Compaq TCP/IP Services for OpenVMS

Concepts and Planning

Part Number: AA-Q06TF-TE

April 2002

Software Version: Compaq TCP/IP Services for OpenVMS Version 5.3

Operating Systems: OpenVMS Alpha Version 7.2–2, 7.3 OpenVMS VAX Version 7.2, 7.3

This manual describes concepts and planning tasks to prepare you to use the Compaq TCP/IP Services for OpenVMS product.

Compaq Computer Corporation
Houston, Texas

Compaq shall not be liable for technical or editorial errors or omissions contained herein. The information is provided "as is" without warranty of any kind and is subject to change without notice. The warranties for Compaq products are set forth in the express limited warranty statements accompanying such products. Nothing herein should be construed as constituting an additional warranty.

Contents

4 OpenVMS Operating System TCP/IP Features

5 Network Server Services

6 Mail Services

7 Connectivity Services

8 Domain Name System/BIND (DNS/BIND)

9 IPv6

Glossary

Index

Figures

Tables

Preface

An open communications standard defined by the worldwide networking community, TCP/IP consists of numerous application, routing, transport, and network management protocols. These protocols enable any connected host to communicate with any other connected host, without needing to know details about the other host or the intervening network topology. Computers and networks from different manufacturers running different operating systems can interoperate seamlessly.

The Compaq TCP/IP Services for OpenVMS product is Compaq's implementation of the TCP/IP networking protocol suite and internet services for OpenVMS Alpha and OpenVMS VAX systems.

This manual introduces the TCP/IP Services product and provides conceptual and planning information to help you configure and manage the product.

Intended Audience

This manual is for anyone who needs an overview of the TCP/IP Services product.

See the *Compaq TCP/IP Services for OpenVMS User's Guide* for information on using TCP/IP Services applications and the *Compaq TCP/IP Services for OpenVMS Management* guide for details on configuring and managing the TCP/IP Services product.

Document Structure

This manual contains the following chapters:

Chapter 1 provides an overview of the TCP/IP Services product.

Chapter 2 describes the network implementation differences between UNIX and OpenVMS.

Chapter 3 describes the many decisions you need to make about OpenVMS configuration options before configuring TCP/IP Services.

Chapter 4 describes OpenVMS operating system features that support the TCP/IP environment.

Chapter 5 describes key concepts of network server features: NTP, routing, BOOTP and DHCP, FTP, and SNMP.

Chapter 6 describes mail services: Post Office Protocol (POP), SMTP, and IMAP.

Chapter 7 discusses ways to connect to the network, such as TELNET, PPP and SLIP, DECnet-over-TCP/IP, NFS, and XDM.

Chapter 8 describes the TCP/IP Services implementation of the Berkeley Internet Name Domain (BIND) service.

Chapter 9 provides guidelines, scenarios, and checklists for deploying IPv6 on a single system in a network.

The Glossary defines terms and acronyms related to TCP/IP Services.

Related Documentation

The following table lists the documents available with this version of Compaq TCP/IP Services for OpenVMS:

Manual	Contents
Compaq TCP/IP Services for OpenVMS Concepts and Planning	This manual introduces TCP/IP Services and provides conceptual and planning information to help you configure and manage the product. This manual also provides a glossary of terms and acronyms, lists the RFCs associated with this product, and documents how to register your network and domain and name servers.
Compaq TCP/IP Services for OpenVMS Release Notes	This text file describes new features and changes to the software, including installation, upgrade, configuration, and compatibility information. These notes also describe new and existing software problems and restrictions, and software and documentation corrections. Print this text file at the beginning of the installation procedure and read it before you install Compaq TCP/IP Services for OpenVMS.
Compaq TCP/IP Services for OpenVMS Installation and Configuration	This manual explains how to install and configure the Compaq TCP/IP Services for OpenVMS product.
Compaq TCP/IP Services for OpenVMS User's Guide	This manual describes how to use the applications available with Compaq TCP/IP Services for OpenVMS, such as remote file operations, e-mail, TELNET, TN3270, and network printing. This manual also explains how to use these services to communicate with systems on private internets or on the worldwide Internet.
Compaq TCP/IP Services for OpenVMS Management	This manual describes how to configure and manage the Compaq TCP/IP Services for OpenVMS product. Use this manual with the *Compaq TCP/IP Services for OpenVMS Management Command Reference* manual.
Compaq TCP/IP Services for OpenVMS Management Command Reference	This manual describes the Compaq TCP/IP Services for OpenVMS management commands. Use this manual with the *Compaq TCP/IP Services for OpenVMS Management* manual.
Compaq TCP/IP Services for OpenVMS ONC RPC Programming	This manual presents an overview of high-level programming using open network computing remote procedure calls (ONC RPC). This manual also describes the RPC programming interface and how to use the RPCGEN protocol compiler to create applications.

Compaq TCP/IP Services for OpenVMS Sockets API and System Services Programming	This manual describes how to use the Sockets API and OpenVMS system services to develop network-based applications.
Compaq TCP/IP Services for OpenVMS SNMP Programming and Reference	This manual describes the Simple Network Management Protocol (SNMP) and the SNMP application programming interface (eSNMP). It describes the subagents provided with TCP/IP Services, utilities provided for managing agents, and how to build your own subagents.
Compaq TCP/IP Services for OpenVMS Management Command Quick Reference Card	This reference card lists the TCP/IP management commands by component and describes the purpose of each command.
Compaq TCP/IP Services for OpenVMS UNIX Command Reference Card	This reference card contains information about commonly performed network management tasks and their corresponding TCP/IP management and Compaq Tru64 UNIX command formats.
Compaq TCP/IP Services for OpenVMS Tuning and Troubleshooting	This manual provides information about how to isolate the causes of network problems and how to tune the TCP/IP Services software for the best performance.
Compaq TCP/IP Services for OpenVMS Guide to IPv6	This manual describes the IPv6 environment, the roles of systems in this environment, the types and function of the different IPv6 addresses, and how to configure TCP/IP Services to access the 6bone network.

For additional information about TCP/IP Services for OpenVMS, access the Compaq OpenVMS World Wide Web site at the following URL:

```
http://www.openvms.compaq.com
```

This manual describes concepts that are specific to the Compaq TCP/IP Services for OpenVMS implementation of TCP/IP. If you are looking for a comprehensive overview of the TCP/IP protocol suite, you might find the following useful:

- Comer, Douglas E. *Internetworking with TCP/IP Volume 1: Principles, Protocols, and Architecture*. 4th edition. Englewood Cliffs, NJ: Prentice Hall; ISBN: 0130183806, 2000.
- Stevens, W. Richard. *UNIX Network Programming Volume 1: Networking APIs: Sockets and XTI*. Second edition, Prentice Hall PTR; ISBN: 013490012X, 1997

Reader’s Comments

Compaq welcomes your comments on this manual. Please send comments to either of the following addresses:

Internet: `openvmsdoc@compaq.com`

Mail: Compaq Computer Corporation
OSSG Documentation Group, ZKO3-4/U08
110 Spit Brook Rd.
Nashua, NH 03062-2698

How to Order Additional Documentation

Visit the following World Wide Web address for information about how to order additional documentation:

`http://www.openvms.compaq.com`

Conventions

The following conventions are used in this manual:

Ctrl/ *x* — Indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.

PF1*x* — A sequence such as PF1*x* indicates that you must first press and release the key labeled PF1 and then press and release another key or a pointing device button.

Return — In an example, a key name enclosed in a box indicates that you press that key.

. . . — A horizontal ellipsis in examples indicates one of the following possibilities:

- Additional optional arguments in a statement have been omitted.
- The preceding item or items can be repeated one or more times.
- Additional parameters, values, or other information can be entered.

⋮ — A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted\par because they are not important to the topic being discussed.

( ) — In command format descriptions, parentheses indicate that you must enclose choices in parentheses if you specify more than one.

[ ] — In command format descriptions, brackets indicate optional choices. You can choose one or more items or no items. Do not type the brackets on the command line. However, you must include the brackets in the syntax for OpenVMS

directory specifications and for a substring specification in an assignment statement.

|

In command format descriptions, vertical bars separate choices within brackets or braces. Within brackets, the choices are optional; within braces, at least one choice is required. Do not type the vertical bars on the command line.

{ }

In command format descriptions, braces indicate required choices; you must choose at least one of the items listed. Do not type the braces on the command line.

Type

This typeface represents the introduction of a new. It also represents the name of argument an attribute, or a reason.

italics

Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER=*name*), and in command parameters in text (where (*dd*) represents the predefined par code for the device type.

UPPERCASE TEXT

Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.

`Monospace text`

Monospace type indicates code examples and interactive screen displays.

In the C programming language, monospace type in text identifies the following elements: keywords, the names of independently compiled externalfunctions and files, syntax summaries, and references to variables or identifiers introduced in an example.

–

A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.

numbers

All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes---binary, octal, or hexadecimal---are explicitly indicated.

1

Introducing Compaq TCP/IP Services for OpenVMS

The Compaq TCP/IP Services for OpenVMS product is the OpenVMS implementation of the industry-standard TCP/IP suite of communications protocols. With TCP/IP Services, users, administrators, and programmers can perform tasks from anywhere in the network, such as:

- Network file access: accessing files on remote hosts
- Sending e-mail: exchanging messages between hosts
- Application development: developing TCP/IP applications for communication between local and remote hosts
- File transfer: exchanging files between hosts
- Accessing user information: accessing information about other users logged onto local or remote hosts
- Remote management: managing and monitoring the network and applications from remote hosts
- TELNET: logging on to a remote host
- Remote command execution: issuing commands to remote hosts
- Remote printing: sending print jobs to a remote printer, and receiving print jobs from a remote host
- Networking booting: providing boot service for a remote host

Users can perform internetworking tasks seamlessly without worrying about the hardware details of each individual network. The TCP/IP Services provides interoperability and resource sharing between OpenVMS systems, UNIX systems, and other systems that support the TCP/IP protocol suite and Sun Microsystems Network File System (NFS). Internet hosts share data and resources by using standard TCP/IP protocols over a number of network hardware configurations including Ethernet, Fiber Distributed Data Interface (FDDI), Token Ring, and asynchronous transfer mode (ATM).

This chapter discusses the following topics. More details about these topics are provided elsewhere in this manual and in other Compaq TCP/IP Services for OpenVMS and OpenVMS documentation set manuals.

- Overview of TCP/IP Services
- Other Compaq OpenVMS products that require TCP/IP
- Application programming interfaces (APIs)
- Requests for Comments (RFCs)

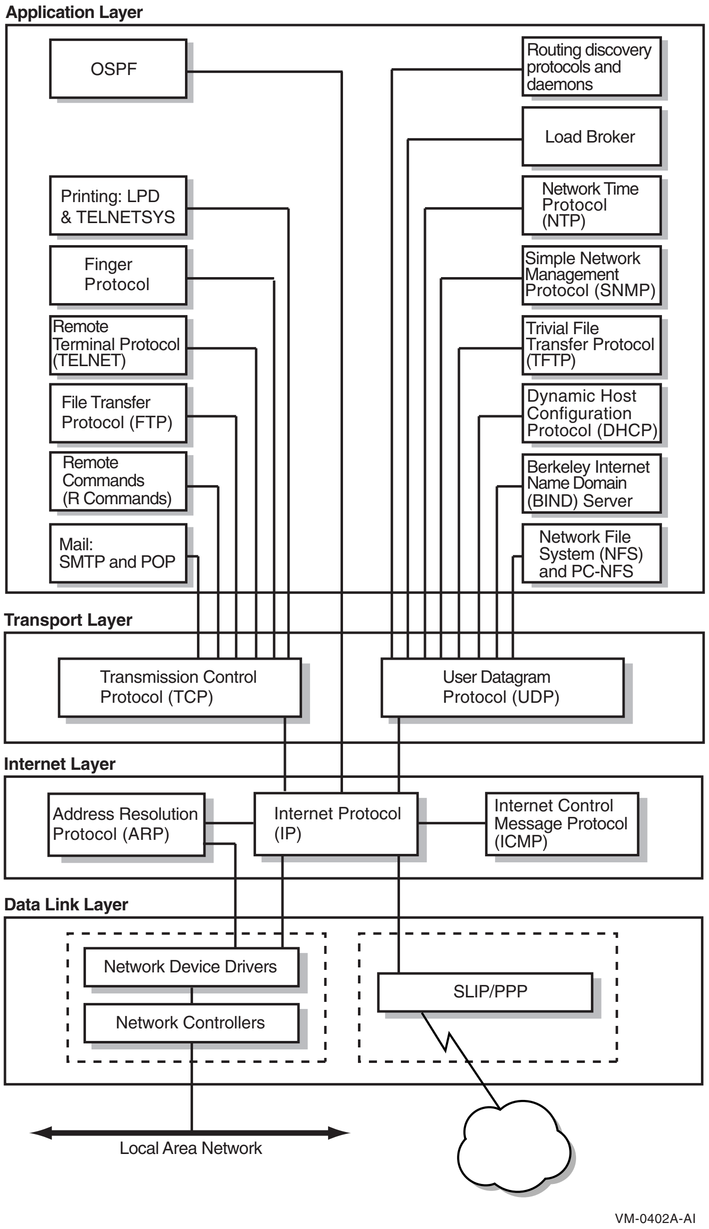
1.1 Overview of TCP/IP Services

TCP/IP Services provides support for several protocols at every level of the TCP/IP model's protocol layers.

- Data Link layer
- Internet layer
- Transport layer
- Application layer

Figure 1–1 shows the various layers and protocols of the TCP/IP model. A description of each layer and protocol follows the figure.

Figure 1–1: The TCP/IP Model


Application Layer
OSPF
Routing discovery protocols and daemons
Load Broker
Printing: LPD & TELNETSYS
Network Time Protocol (NTP)
Finger Protocol
Simple Network Management Protocol (SNMP)
Remote Terminal Protocol (TELNET)
Trivial File Transfer Protocol (TFTP)
File Transfer Protocol (FTP)
Dynamic Host Configuration Protocol (DHCP)
Remote Commands (R Commands)
Berkeley Internet Name Domain (BIND) Server
Mail: SMTP and POP
Network File System (NFS) and PC-NFS
Transport Layer
Transmission Control Protocol (TCP)
User Datagram Protocol (UDP)
Internet Layer
Address Resolution Protocol (ARP)
Internet Protocol (IP)
Internet Control Message Protocol (ICMP)
Data Link Layer
Network Device Drivers
SLIP/PPP
Network Controllers
Local Area Network
VM-0402A-AI

1.1.1 Data Link Layer

At the base of the TCP/IP layers, the Data Link layer formats data and provides services that directly access the physical network.

This layer also receives data that is routed from the Internet layer and transmits the data to its destination, converting logical IP addressesto physical addresses of the network adapter or network interface cards (NICs) using the Address Resolution Protocol (ARP).

Some commonly used network architectures designed for the physical network are Ethernet and its variants, Token Ring, FDDI, and ATM.

A single host computer can have multiple NICs. This configuration is termed a **multihomed** host. Depending on your network setup, the Data Link layer's configuration may vary. For more information, see Chapter 3.

1.1.2 Internet Layer

The Internet layer moves data around the internet. The Internet Protocol uses addressing to deliver and route packets across networks independently of the network cabling.

At this level, the protocols are:

- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)
- Address Resolution Protocol (ARP)

The protocol also encapsulates datagram headers, sends ICMP error and control messages, and maps ARP address conversions.

Routing protocols at this layer are:

- Routing Information Protocol (RIP) Versions 1 and 2
- Open Shortest Path First (OSPF) Version 2
- Exterior Gateway Protocol (EGP)
- Border Gateway Protocol (BGP)
- Router Discovery

For more information about these protocols, see Chapter 5.

1.1.3 Transport Layer

The Transport layer enables a flow of data between two hosts. The protocols at this layer are either **connection oriented**, in which the protocol establishes and maintains the connection between communicating hosts to prevent errors, or **connectionless**, in which a one-way datagram is sent to a destination host.

The TCP/IP Services supports both transport protocols:

- The connection-oriented protocol, Transmission Control Protocol (TCP) provides a reliable data flow between two hosts. TCP is used for complex, large packets that have an IP address.
- The connectionless protocol, User Datagram Protocol (UDP) provides a much simpler service to the Application layer than TCP but does not guarantee reliability. UDP is used for simple, small packets and requests for dynamic IP address assignment. UDP packets have a MAC address.

1.1.4 Application Layer

The top layer of the TCP/IP protocol suite, the Application layer handles the details of the particular application, protocol, or user command; it is not concerned with the movement of data across the network.

TCP/IP Services supports the following TCP/IP applications, protocols, and user services:

Remote Computing Services

Remote computing applications enable networked users to run software on remote systems. TCP/IP Services include the following remote computing application components:

- **TELNET** enables remote login to other hosts in the network. Compaq TCP/IP Services provides simultaneous multiple sessions, IBM3270 terminal emulation (TN3270) and two interface formats: DCL style and UNIX style.
- **Remote,** or **R,** commands are use for the following:
  - RLOGIN for remote login
  - RSH for remote shell capabilities
  - REXEC to execute commands to a remote host
  - RMT/RCD to read magnetic tapes or CD-ROMs from remote hosts
- **XDM** is a network-based graphics window system based on the client/server application model. It enables a system to display information output from an application that is running on another system in the network.
- The **FINGER** utility is used to display user information for the network.

File Transfer Services

TCP/IP Services includes the following components that let users transfer data files between local and remote hosts:

File Transfer Protocol (FTP) transfers files between hosts.

Trivial File Transfer Protocol (TFTP) downloads and transfers files. Compaq TCP/IP Services supports downloading of system image and other information to client hosts.

Resource Services

Line printer/line printer daemon (LPR/LPD) provides printing services to local and remote hosts.

TELNET print symbiont (TELNETSYM) provides remote printing services that enable OpenVMS printing features not available with the LPR/LPD print service.

Network File System (NFS) is a protocol that allows computers to access remote files as if they were local files, regardless of operating system, hardware type, or architectural differences between the local and remote systems. This is accomplished in a client/server environment where specific implementations on NFS exist on both the client and server machines.

PC-NFS is a daemon that enables access to the NFS server from a PC by providing authentication services to PC-NFS clients.

Electronic Mail Services

Communication functions such a electronic mail are vital both within an organizational internet and across the Internet worldwide. The electronic mail components of TCP/IP Services are:

Simple Mail Transfer Protocol (SMTP) is the TCP/IP standard protocol for transferring electronic mail messages from one system to another.

IMAP is the Internet Message Access Protocol. IMAP enables clients to access email messages and folders from an IMAP server and synchronize them locally. This enables a client to organize email messages and folders without continuous access to the server.

Post Office Protocol (POP) is a mail repository used primarily by PCs.

1.2 Application Support

Beyond the industry-standard TCP/IP application services, TCP/IP Services provides support for the following Compaq products:

- PATHWORKS or Advanced Server
- DECnet-Plus

1.2.1 PATHWORKS and DECnet-over-TCP/IP Support

The Compaq TCP/IP Services for OpenVMS software includes the PWIP driver and the PWIPACP network ancillary control process (ACP). The PWIP driver enables communication between OpenVMS systems running both PATHWORKS server and TCP/IP Services software, and personal computers running PATHWORKS client software. It also enables the DECnet-over-TCP/IP feature, which is included with the DECnet-Plus for OpenVMS Version 6.0 and later software. For more information, see Chapter 7.

1.3 APIs

Network applications specific to the Compaq TCP/IP Services can use the following application programming interfaces (APIs):

- Berkeley Sockets Interface
- OpenVMS QIO System Services interface
- ONC RPC programming interface
- SNMP programming interface

1.3.1 Berkeley Sockets Interface

Sockets have become a popular programming interface. The Berkeley Sockets Interface is a programming interface that provides applications with access to network communication protocols. A socket is a generalized UNIX communication endpoint on which the TCP/IP protocols have been implemented. Using the socket programming interface makes it easy to implement network applications.

OpenVMS provides support for the socket interface through the C programming language and the Compaq C Run-Time Library. The benefits of using the socket interface on the OpenVMS platform include:

- Ease of porting network applications from other platforms to the OpenVMS platform. A sockets interface can be generic.

- Many application developers are familiar with the programming environment.
- In addition to the TCP/IP protocols, there are options for other types of protocols.

For more details, refer to the *Compaq TCP/IP Services for OpenVMS Sockets API and System Services Programming* manual.

1.3.2 OpenVMS QIO System Service Interface

The standard I/O programming interface on OpenVMS uses the QIO (queue input/output) system services. QIO services provide a rich set of functions for controlling devices, and connections and for performing input (read) and output (write) operations.

The benefits of using the OpenVMS QIO interface include:

- Support for the QIO interface in the following programming languages:

  MACRO-32
  Compaq C
  Compaq Fortran
  Compaq Ada
  Compaq and VAX BASIC
  VAX BLISS-32
  Compaq COBOL
  VAX Pascal
  Compaq PL/1

- Ability to handle complex applications with many concurrent connections
- Efficient input/output operations
- Robust asynchronous event handling (While sockets offer the ability to do nonblocking I/O operations, they do not offer the ability to perform asynchronous I/O.)
- Ease of DECnet applications portability to TCP/IP protocols

For more details, refer to the *Compaq TCP/IP Services for OpenVMS Sockets API and System Services Programming* manual.

SRI QIO Compatibility

TCP/IP Services provides support for customer applications using the INETDRIVER QIO interface developed at Stanford Research Institute (SRI) in 1980-81. An SRI QIO emulator that translates non-TCP/IP Services QIO interfaces into TCP/IP Services QIO programming interfaces can be configured by using the TCPIP$CONFIG procedure.

1.3.3 ONC RPC Programming Interface

The RPC programming interface is an industry-standard, portable API that is an efficient alternative to using sockets for application development. Programmers do not need an in-depth knowledge of networking protocols to use RPC.

One strong point of the RPC interface is its ability to distribute functions across the network. This is done in an architecture-independent manner, thereby avoiding problems with floating-point formats and byte-address ordering that often occur when interacting between architectures.

This API includes:

- Library of RPC function calls
- **Portmapper service**, which is a service that client programs can use to determine the port number that another service uses. Clients use the Portmapper Service for NFC, PC-NFS, and RPC applications.
- External data representation (XDR) routines

For more details, refer to the *Compaq TCP/IP Service ONC RPC Programming* manual.

1.3.4 SNMP Programming Interface

The Extensible Simple Network Management Protocol (eSNMP) API provides routines for developing applications that remotely manage and collect data from network devices such as routers, bridges, and hosts.

These network devices run software that carries out management commands that either get information from devices or set operating parameters for devices.

Other network applications send commands to network devices to perform configuration management, monitor network traffic, or troubleshoot network problems.

The SNMP API provides routines for the following functions:

- Establish, maintain, and terminate communication with the master agent
- Manipulate, reformat, extract, and compare data
- Control information that is written to log files

The SNMP API routines are almost identical in function and interface with the routines in the Compaq *Tru64* UNIX API.

For more details, refer to the *Compaq TCP/IP Services for OpenVMS SNMP Programming and Reference* manual.

1.4 Understanding RFCs

Although TCP/IP is monitored by a number of organizations, no single entity owns this protocol; its specifications are publicly available and are constantly growing as communications requirements evolve.

The process by which the specifications evolve is through a mechanism called **Requests for Comments** (**RFCs**). When someone has an idea for a new or improved capability for TCP/IP, he or she writes a proposal, posts it on the Internet as an Internet draft, and requests comments from the networking community. After a review and revision cycle, working code is developed and an RFC becomes a standard protocol.

RFCs are available on the Internet from the Internet Network Information Center (InterNIC). The following web site provides links to several RFC international repositories, lists all RFCs, and explains how you can obtain copies:

```
http://www.rfc-editor.org
```

Note that, although RFCs recommend implementation guidelines, the actual implementation of an RFC can and must differ from the RFC in minor ways. When product documentation refers to specific RFCs, be aware that the RFC only sets the standard for development. Product developers must design their software for the specific environment and requirements of their customers.

2

Understanding OpenVMS and UNIX Implementations

An important step in planning a network host implementation is to gain an understanding of the computing environments in which the network services will run. Compaq *Tru64* UNIX implementations of TCP/IP Services are often ported to OpenVMS. As a result, they often appear to be identical. However, there are many significant differences. This chapter describes key implementation differences between UNIX and OpenVMS networks. The following topics are discussed:

- Evaluating the computing environment
- File compatibility
- Portability
- Determining which file system to use

Things to Consider

In planning your TCP/IP Services environment, consider the following:

- Do I need to migrate from one operating system to the other?
- Do I need to set up a system that coexists with multiple operating systems?
- Should I choose a Files–11 file system or a container file system?

2.1 Evaluating the Computing Environment

The issues of working in a heterogeneous computing environment that includes OpenVMS and Tru64 UNIX operating systems are complex. Consider these concepts when evaluating or implementing an interoperability strategy:

- **Migration** occurs when software applications are rewritten as necessary to be ported from one operating system to the other. In a general sense, not only the applications but also the users migrate to another system. Migration implies a gradual replacement of the original system with the new system.
- **Coexistence** occurs when two or more systems, such as OpenVMS and Tru64 UNIX, are maintained as part of a larger, heterogeneous computing environment. The amount of interoperability varies with the individual configurations. It is possible to set up nearly identical network configurations, thereby reducing maintenance.

2.1.1 Understanding the Open Systems Concept

The client/server model of computing means that users running applications on their PCs and workstations are networked with larger departmental systems. The departmental systems provide a variety of services to the clients, such as access to common database, print, and backup/archive services.

To best serve this model of computing, the systems must be open. Open environments support interoperability and application portability and enable developers and users to easily use different platforms. The OpenVMS operating system is an **open system** with an extensive list of functions.

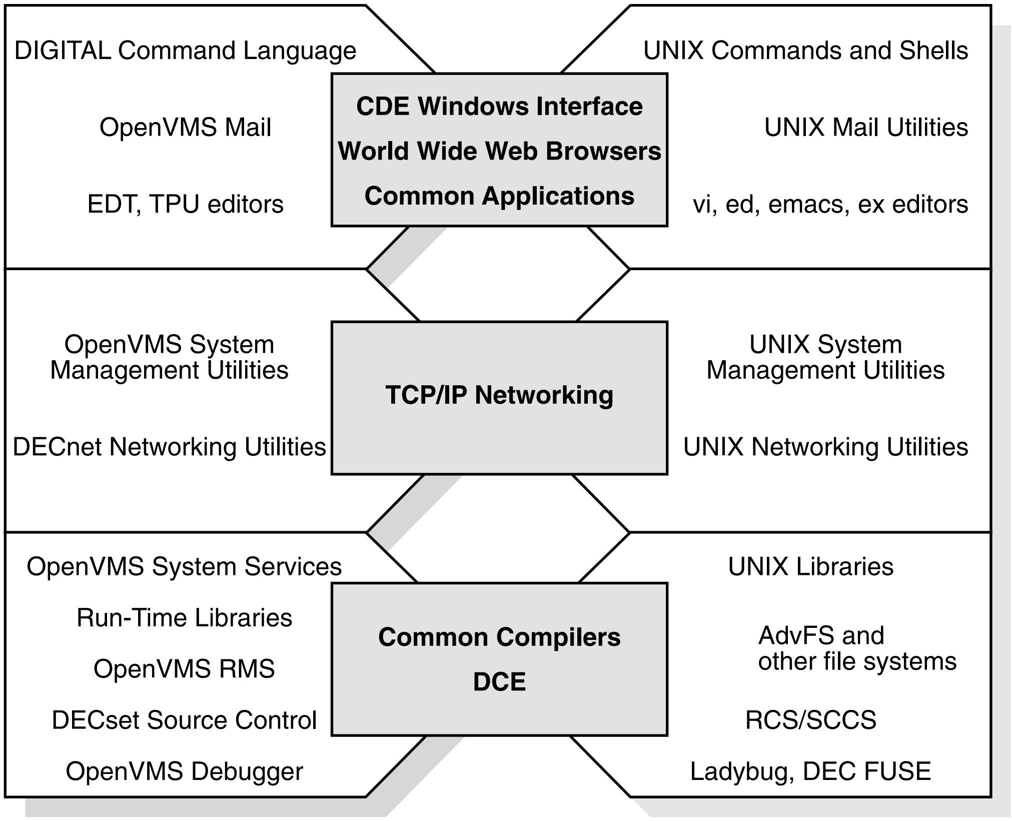
An open system allows the OpenVMS operating system, whether powered by Alpha or VAX, to interoperate efficiently with Compaq Tru64 UNIX and with other vendors' operating systems, particularly with Windows NT and other UNIX operating systems.

2.1.2 Understanding the Middleware Concept

Implementing open systems means using the right **middleware** between the operating system and the hardware platform and applications. Consistent middleware affords interoperability and portability when and where it is needed. An open systems strategy involves a consistent middleware approach that is based on standards and is embodied in layered software and in individual operating systems, such as OpenVMS and UNIX. **Compaq DCE** is an example of middleware, with standard interfaces supported on both OpenVMS and UNIX. Compaq DCE, or Distributed Computing Environment, is an architecture of standard programming interfaces, conventions, and server functions that transparently distributes applications across heterogeneous networks.

As shown in Figure 2-1, OpenVMS and UNIX interoperate efficiently, especially in areas that are common to both platforms: windowing interface, standard POSIX and DCE programming interfaces, compilers, networking products, and applications.

Figure 2–1: Comparison of OpenVMS and UNIX



VM-0896A-AI

2.2 File Compatibility

The **Network File System**, or (NFS) provides a standard for the exchange of data between machines running different operating systems. NFS enables users to access directories and files on remote computers transparently, as if they were on the local system. NFS accomplishes this because it is implemented on the both the remote and the local computer.

NFS protocol achieves portability between different machines, operating systems, network architectures, and transport protocols through the use of Remote Procedure Call (RPC) and External Data Representation (XDR). For more information about RPCs and XDR, refer to the *Compaq TCP/IP Services for OpenVMS ONC RPC Programming* manual.

Using NFS is simple. Configuring and implementing NFS, however, are more complex. For NFS concepts and considerations, as well as detailed configuration and implementation information, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

TCP/IP Services accommodates the numerous key differences between UNIX and OpenVMS to make user interaction between the two operating systems appear transparent. This enables all systems on a heterogeneous network to store and share files and applications regardless of file specification and structure differences.

This section discusses:

- Directory hierarchies
- File specifications
- Linking files
- File structures
- File ownership
- File protection
- UNIX style file system on TCP/IP Services hosts

2.2.1 Directory Hierarchies

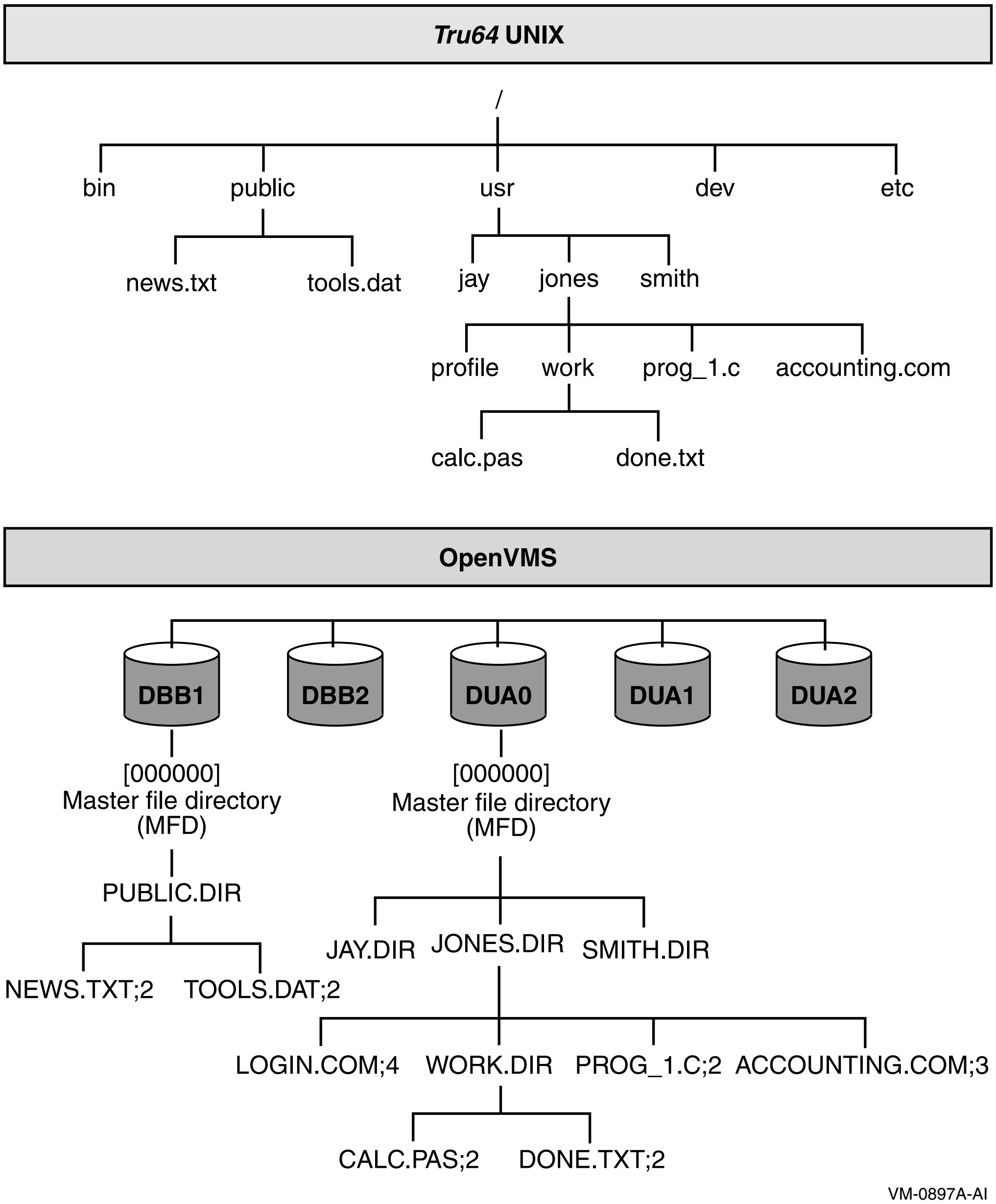
Unlike OpenVMS, the UNIX hierarchy appears as one tree (starting from the root directory, or “/”) that can be located on more than one device. Table 2-1 describes the differences between the OpenVMS and Tru64 UNIX directory hierarchies.

Table 2–1: Directory Hierarchy Differences

OpenVMS	UNIX
Reside on one volume having one root above all directories on the volume.	Can reside on multiple volumes.
Device names included in file specifications.	Device names not included in file specifications.

Figure 2-2 illustrates the differences between a UNIX file structure and an OpenVMS file structure.

Figure 2–2: Comparison of UNIX Directory and OpenVMS Directory Hierarchies



2.2.2 File Specifications

There are basic differences between OpenVMS and UNIX file specifications. Table 2–2 summarizes the differences.

Table 2–2: File Specification Differences

OpenVMS	UNIX
Files are delimited in the following way: • A colon (:) separates the device from the directory. • Square brackets ([ ]) or angle brackets (< >) enclose the directory and any subdirectories. • 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. The ODS-5 file system implements extended file specifications and is a step toward improving interoperability. ODS-5 is described later in this chapter. For complete details about the ODS-5 file specification, refer to the OpenVMS product documentation.	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. Subsequent slashes separate each element of 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.

OpenVMS file specification format

On a standard **Files-11 On-Disk Structure Level 2 (ODS-2)** volume, an OpenVMS file specification has the following format:

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

UNIX file specification format

On a UNIX system, the file specification has the following format:

```
/directory/subdirectory/filename
```

2.2.3 Absolute and Relative File Specifications

OpenVMS and UNIX both have two types of file specifications or pathnames: absolute and relative. Table 2–3 describes the differences between the two platforms.

Table 2–3: Absolute and Relative File Specification Differences

OpenVMS	UNIX

The relative path for file `calc;1` in directory `usr:[jones]` is:
`[.accounting.calc;1]`
The absolute path is:
`usr:[jones.accounting.calc;1]`
	The relative pathname for file `calc` in directory `/usr/jones` is
	`accounting/calc`
	The absolute pathname is
	`/usr/jones/accounting/calc`
	On UNIX systems, absolute pathnames use the entire directory path that leads to the file, beginning with the root, which is represented by an initial slash.
	The root directory is the first directory in the file system. All other files and directories trace their ancestry back to the root. Relative pathnames begin the directory path with the current working directory and exclude the current working directory name in the pathname. There is no initial slash in a relative pathname.

2.2.4 File Specifications

There are fundamental differences between file names specified in OpenVMS and in UNIX. Table 2–4 describes those differences.

Table 2–4: File Specification Differences

OpenVMS (ODS-2)	UNIX

Includes, in this order:
1. the file name
2. the file type
3. an optional version number
An OpenVMS file specification can have a maximum of 255 characters.
The file name and file type can have up to 39 characters each and are separated by a period. For example:
`FILE_NAME.TXT;1`
Valid characters in an OpenVMS file name or type include: A–Z, a–z, 0–9, underscore (_), hyphen (-), and dollar sign ($). The version number (preceded by a semicolon) is a decimal number from 1 to 32767; it differentiates versions of the same file.
	Contains up to 1024 characters, with each element of the pathname containing up to 255 characters. UNIX file specifications have the following format:
	`file_name.txt`
	Some older versions of the UNIX operating system limit the size of one element to 14 characters, or have other limits that you can change if you recompile the kernel.
	In theory, you can use any ASCII character in a UNIX pathname except for the slash (/) and null characters. For example, a valid file name in UNIX can be:
	`report.from.january_24`
	However, avoid characters (such as the pipe (|) character) that can have special meaning to the UNIX shell.

2.2.5 Case Sensitivity

Case sensitivity differs between the two operating systems. Table 2–5 describes the difference.

Table 2–5: Case-Sensitivity Differences

OpenVMS (ODS-2)	UNIX
Stores everything in uppercase. For example, any case variations of the following file name is stored in uppercase: CHAPTER_ONE.TXT;1	Regards uppercase and lowercase characters as different characters.
	For example, on a UNIX system, the following file names represent three different files: • CHAPTER_ONE.TXT • Chapter_One.Txt • chapter_one.txt

2.2.6 File Types

Table 2–6 describes the file type differences between OpenVMS and UNIX.

Table 2–6: File Type Differences

OpenVMS	UNIX
Important in OpenVMS file identification The file type usually describes the kind of data in the file.	UNIX systems do not use file types. However, UNIX has certain naming conventions that resemble OpenVMS file types.
For example, a text file typically has a file type of .TXT.	For example, file names ending in `txt` are text files.
All OpenVMS directories have a file type of .DIR.	UNIX directories do not have file types.

2.2.7 Version Numbers

Table 2–7 describes file version number differences between OpenVMS and UNIX.

Table 2–7: Version Number Differences

OpenVMS	UNIX
Every 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 when subsequent 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.	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.
For example: FILE_NAME.TXT;1 FILE_NAME.TXT;2 FILE_NAME.TXT;3	For example: `file_name.txt`

2.2.8 Linking Files

A link is a directory entry that refers to a file or another directory. Table 2–8 describes the differences between OpenVMS and UNIX file linking.

Table 2–8: Link Files Differences

OpenVMS	UNIX
Files can exist without links.	Files cannot exist without links.
Hard Links OpenVMS systems allows you to perform a function similar to hard links with the SET FILE/ENTER and SET FILE/REMOVE commands. The OpenVMS operating system does not maintain a count of links to a file. As a result, you can delete a file without deleting its links.	**Hard Links** Hard links allow users to share the same file under different pathnames. A hard link cannot span file systems. On UNIX systems, any changes to the file are independent of the link used to refer to the file. The UNIX system maintains a count of the number of links to each file. If removing a link results in the link count becoming zero, the file is deleted. A file can be deleted only by removing all of its links.
Symbolic Links OpenVMS file systems do not support symbolic links.	**Symbolic Links** A symbolic link is a file that contains the name of the file to which it is connected. Symbolic links provide a path to the original file. A UNIX symbolic link can span file systems. Unlike a hard link, a symbolic link does not maintain a link count. In addition, symbolic links can exist after the file is deleted. However, the system returns an error if the symbolic link file is accessed after the file it names is deleted.

2.2.9 File Structures

Table 2–9 describes the differences between the OpenVMS and UNIX file structures.

Table 2–9: File Structure Differences

OpenVMS	UNIX
Supports three file structures: indexed, relative, and sequential. OpenVMS also supports the following record formats and record attributes: • Fixed length • Variable length • Variable with fixed-length control (VFC) • Stream (including STREAM_LF and STREAM_CR) • Undefined • Carriage return/carriage control • Fortran carriage control • VFC carriage control	Supports byte streams only. The records in UNIX text files have the same format as the OpenVMS Record Management Services (RMS) STREAM_LF record format.

2.2.10 File Ownership

The OpenVMS and UNIX operating systems use different mechanisms for file ownership. Table 2–10 describes those differences.

Table 2–10: File Ownership Differences

OpenVMS	UNIX
The OpenVMS operating system controls file ownership and access through a user identification code (UIC). A UIC is a 32-bit value that consists of a 14-bit group number, a 16-bit member number, and 2 reserved bits. 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 accessing process and the UIC of the object (the file or directory). OpenVMS controls file access through an access control list (ACL). You can deny or grant read, write, execute, delete, and control access to a user or group of users who have the identifier specified by the ACL. For additional ACL information, refer to the OpenVMS documentation set. The NFS protocol does not provide ACL support. Therefore, the NFS client is unaware of ACLs that the NFS server applies to the file. Consequently, the NFS client cannot use an ACL to control file access. Access control is determined through standard file protections. For more information, see Chapter 2.	The UNIX operating system controls access to files with user identification (UID) and group identification (GID). Tru64 UNIX uses 32-bit UIDs and GIDs. For compatibility, NFS also recognizes 32–bit UIDs and GIDs.

2.2.11 File Protections

The OpenVMS and UNIX operating systems use similar file protection schemes, as shown in Table 2–11.

Table 2–11: Comparison of File Protection

Mechanism	OpenVMS	UNIX
User classifications	SYSTEM (S) OWNER (O) GROUP (G) WORLD (W) Classification depends on the relationship between the UIC of the accessing process and the object.	`user (u)` --- The user has a matching UID `group (g)` --- The user has a matching GID `other (o)` --- Any other user System category is not used; system administrators always have access to UNIX files.

Table 2–11: Comparison of File Protection (cont.)

Protection levels	READ (R) WRITE (W) EXECUTE (E) – Controls file execution and directory search access DELETE (D)	`read (r)` — The user has a matching UIC `write (w)` — Controls unlinking files to the directory. `execute (x)` — Controls file execution and directory search access A file is deleted if it is unlinked from the directory and had no links in other directories. Write access to the directory is refused.
Syntax	`s:rwed, o:rwed, g:rwed, w:rwed`	`rwxrwxrwx` The protection levels are divided into groups of three characters: • First three characters: protection levels for the owner. • Second three characters: protection levels for the group. • Last three characters: protection levels for all other users.

2.3 Portability

The TCP/IP Services allows you to create a logical UNIX style file system on an OpenVMS host. Remote UNIX hosts that have NFS software can then access this file system. When a remote UNIX system accesses files, these files conform to UNIX file system rules, not to the OpenVMS rules. This ensures that existing UNIX applications work without change. The logical UNIX file system resides on a Files-11 formatted disk and is represented as a set of Files-11 files called a **container file system**. For information about creating a UNIX file system on an OpenVMS host, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

The UNIX file names and attributes are catalogued in the container file, one of the files in the container file system. The container file also has a representation of the UNIX directory hierarchy and a pointer to the data file for each file name. In addition to its UNIX name, each file in the container file system has a valid Files-11 file name assigned by the system. An OpenVMS directory exists for each UNIX directory stored in the container file. All files catalogued in a UNIX directory are also catalogued in the corresponding OpenVMS directory. However, the UNIX directory hierarchy is not duplicated in the OpenVMS directory hierarchy. Each UNIX file is represented as an OpenVMS data file. Therefore, OpenVMS utilities, such as BACKUP, can use standard methods to access these files.

2.4 Determining Which File System to Use

The first step in managing your TCP/IP Services system is to decide which file system to use. NFS on OpenVMS enables you to set up and export three different kinds of file systems:

- **OpenVMS On-Disk Structure (ODS-2) file system**, in which devices, directories, and files are stored on a Files-11 formatted disk
- **OpenVMS On-Disk Structure (ODS-5) file system**, which enables creation and storage of files with extended file names for compatibility with other file systems, such as Windows.
- **UNIX,** or **container, file system,** built on top of an OpenVMS system. If you are not familiar with OpenVMS file systems, refer to the *OpenVMS System Manager's Manual: Essentials* to learn how to set up and initialize a Files-11 disk. As Figure 2–2 shows, both file systems are structured as hierarchical, multilevel directories. On OpenVMS systems, the top level is called the **master file directory,** or **MFD**. This directory contains all the directories and reserved system files. The directory is named [000000]. On UNIX systems, the top-level directory is called the root, or / .

Table 2–12 lists the NFS server features available to non-OpenVMS clients based on file system choice.

Table 2–12: NFS Server Features Available to Non-OpenVMS Clients

Features	ODS-2	OD2–2 with name conversion	ODS-5	Container file system
Files easily shared between remote clients and local OpenVMS users	Yes	Yes	Yes	No
Mixed case, special characters, and extra dots in file names	No	Yes	Yes	Yes
Long file names	No	No	Yes	Yes
File names look the same to remote clients and local OpenVMS users	Uppercase to local users, lowercase to remote clients	No	Yes	N/A
Support for hard links, symbolic links, special files	No	No	No	Yes
UNIX compatible timestamps	No	No	No	Yes
Case-sensitive lookup	N/A	Yes	No	Yes

The dual cataloguing of files to both OpenVMS file systems limits the set of DCL commands. OpenVMS utilities, such as BACKUP, can use standard methods to access the files. However, except for backing up and restoring files, you should not use DCL commands to manipulate files in a container file system.

Decision Point

Your file system choice depends on your environment and the user needs on the NFS client host. Consider using an OpenVMS file system if:

□ Your users share most files between your OpenVMS system and another OpenVMS host, or between your OpenVMS system and a UNIX client.

□ Your client users need to maintain multiple versions of files.

□ You share files between users on OpenVMS and users on NFS clients.

□ File sharing between your OpenVMS system and a UNIX client is minimal.

□ Client applications use symbolic or hard links or special files.

For More Information

For more information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Management* manual:

- Setting up container file systems
- Configuring and implementing the NFS server

For a list of commonly used Tru64 UNIX commands and their equivalents on OpenVMS, refer to the *Compaq TCP/IP Services for OpenVMS Tuning and Troubleshooting* manual.

For more details about interoperability between UNIX and OpenVMS, refer to the *OpenVMS and Compaq UNIX Interoperability and Migration Guide*. This guide discusses products and services, available both from Compaq and from other vendors, that might provide solutions to interoperability problems.

For more information about RPCs and XDR, refer to the *Compaq TCP/IP Services for OpenVMS ONC RPC Programming* manual.

For additional ACL information, refer to the OpenVMS documentation set.

3

OpenVMS Server and Network Configurations

There are several server and network configurations to consider before installing TCP/IP Services for OpenVMS. This chapter describes the following concepts that will enable you to make informed decisions about these configuration options:

- OpenVMS VAX and Alpha similarities and differences
- Cluster environments
- Multiple interfaces and multihoming
- Pseudointerfaces
- Serial lines

Note

VAX development has limited continued support. VAX users should consider migrating to Alpha, if possible.

Things to Consider

In planning your TCP/IP Services for OpenVMS configurations, consider the following:

- Does the network contain VAX or Alpha systems, or both?
- Is my system running a DHCP server?
- Is my system running a DHCP client?
- How many interfaces does the system have?
- Do I have serial lines in my network? If so, for which systems are they used?

3.1 Understanding OpenVMS VAX and Alpha Systems

You need to consider several issues when you plan to add one or more OpenVMS Alpha systems to your OpenVMS VAX computing environment. For full details about the similarities and differences between OpenVMS Alpha and OpenVMS VAX, refer to the *OpenVMS Compatibility Between VAX and Alpha* guide, which is available on line at:

```
http://www.openvms.compaq.com/doc
```

3.1.1 User Environment

The user environment on OpenVMS Alpha is virtually the same as that on OpenVMS VAX. Table 3–1 describes the similarities and differences.

Table 3–1: OpenVMS VAX and OpenVMS Alpha Similarities and Differences

Component Similarities	OpenVMS VAX Differences	OpenVMS Alpha Differences
DIGITAL Command Language (DCL) Essentially the same on both systems.	None	Refer to the few exceptions in the *OpenVMS Compatibility Between VAX and Alpha* guides available on line.
DCL Help Most DCL help text is common to both systems.	System-specific information is identified by the phrase “On VAX.”	System-specific information is identified by the phrase “On Alpha.”
DCL command procedures Most DCL command procedures, with commands, qualifiers, and lexical functions, work on both systems.	None	A few command procedures contain qualifiers not available on OpenVMS Alpha.
Databases Standard databases, such as Oracle Rdb, function the same on both systems.	None	Most third-party databases available for OpenVMS VAX are also available for OpenVMS Alpha. For more information, refer to the *OpenVMS Compatibility Between VAX and Alpha* guide, available on line.

3.1.2 System Management Environment

Most OpenVMS VAX system management utilities, command formats, and tasks are identical on OpenVMS Alpha, with the following exceptions:

- On VAX, use of the POLYCENTER Software Installation utility is limited to the installation of layered products, such as Compaq TCP/IP Services for OpenVMS.
- On Alpha, the POLYCENTER Software Installation utility is also used to install both the OpenVMS operating system and layered products.

For more information about implementation differences between OpenVMS VAX and OpenVMS Alpha, refer to the *OpenVMS System Manager’s Manual*.

3.1.3 Programming Environment

The same types of programming development tools that OpenVMS VAX programmers use are available on OpenVMS Alpha systems, such as the Linker utility, the Librarian utility, the OpenVMS Debugger (also known as the symbolic debugger), the Delta/XDelta Debugger, and run-time libraries. However, some TCP/IP Services components are available only on OpenVMS Alpha, including:

- BIND Version 9
- IMAP
- PPP

These components are introduced later in this manual.

For details about the similarities and differences between the programming environment on VAX and Alpha, refer to *A Comparison of System Management on OpenVMS AXP and OpenVMS VAX*, which provides guidelines for developing applications that run on both OpenVMS VAX and OpenVMS Alpha, as well as additional guidelines for systems that run in a mixed-architecture OpenVMS Cluster.

3.2 OpenVMS Cluster Configuration

Compaq TCP/IP Services for OpenVMS supports OpenVMS Cluster systems and the use of cluster aliases. The network sees the cluster as one system with one name, the **cluster alias**. A remote host can use the cluster alias to address the cluster as one host, or it can use the host name of a cluster member to address a cluster member individually.

In a DECnet network, it is convenient to be able to treat nodes within a homogeneous OpenVMS Cluster as though they were a single node. You can do this by establishing an **alias node identifier** for the cluster. You can specify the alias node identifier as either a unique node address or a corresponding node name. Any member node can elect to use this special node identifier as an alias while retaining its own unique node identification. For more information on the use of the optional cluster alias node identifier, refer to the *DECnet for OpenVMS Networking Manual*.

Note

> DECnet–Plus software is not required in an OpenVMS Cluster configuration. However, DECnet–Plus is necessary if internode process-to-process communication using DECnet mailboxes is needed. For more information about DECnet-Plus in an OpenVMS Cluster configuration, refer to the *Guidelines for OpenVMS Cluster Configurations* manual.

For load balancing, an OpenVMS Cluster can consist entirely of OpenVMS Alpha nodes or of a combination of OpenVMS VAX and OpenVMS Alpha nodes.

You can have numerous OpenVMS Cluster configurations. For complete information about supported devices and configurations, refer to *Guidelines for OpenVMS Cluster Configurations* and the *OpenVMS Cluster Software Software Product Description (SPD)*. For complete information about setting up and using an OpenVMS Cluster environment, refer to the *OpenVMS Cluster Systems* manual.

3.2.1 Failover Capability

Failover capability is the hallmark of a cluster environment. If one computer , or node, in the cluster fails, the others can assume its functionality and continue. This is called **automatic failover**.

Each **node** (as a member of the host configuration in the cluster) retains a separate IP address. This is beneficial for troubleshooting the individual node because you can `ping` the specific node to see whether it is running.

All of the TCP/IP services support automatic failover and can run on multiple nodes in an OpenVMS Cluster. For example, if more than one node in the cluster is running the NFS server, the cluster can appear to the NFS client as a single host. For more information about configuring a specific service for cluster failover, refer to the particular service in the *Compaq TCP/IP Services for OpenVMS Management* guide.

3.2.2 Connection Load Balancing

Load balancing using the TCP/IP Services is defined by the **load broker**. The load broker is a configurable, calculated, load-balancing mechanism for distributing the work load among **DNS** (Domain Name System, which maintains and distributes information about Internet hosts) cluster members. For more information about DNS, see Chapter 5.

Unlike **round-robin scheduling** (the default method used by most DNS name servers, in which each individual node in the cluster is polled in a continual, specific order), the load broker takes into account the load on all DNS cluster participants. The load broker polls DNS cluster members and updates the metric server accordingly.

When the load broker starts, it reads its configuration file and starts polling DNS cluster members. The load broker exchanges messages with DNS cluster members that run the **metric server**, which calculates the current load on a DNS cluster host by using a specific equation. The metric server calculates the current rating and reports it when polled by the load broker. Periodically, the load broker sorts the list of addresses based on metric rating reports, drops the systems that do not respond after being polled three times, and compares a subset of the list with the name server information.

To do the comparison, the load broker sends a host lookup request to the specified name server. If the lists are the same, the load broker does not make changes. If the lists are different, the load broker updates the name server data by sending a dynamic update request to the specified name server. The name server uses round-robin scheduling to further balance the load across the members of a DNS cluster. Thus, every consecutive request for translating the DNS cluster name results in a returned list, is rotated by one.

For specific information about configuring the load broker, starting and stopping the metric server, and troubleshooting, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

3.3 Multihoming and Multiple Interfaces

Although host computers can have several network interface cards (NICs) installed, you can configure the host through a single, primary interface. This section introduces the following concepts:

- Multihomed computers
- Primary interface
- Pseudointerfaces

3.3.1 Multihomed Computers

Individual host computers can have multiple network interface cards per computer. Such a computer is called **multihomed**. These physical interfaces can be connected to different types of networks, such as Ethernet, FDDI, Token Ring, asynchronous transfer mode (ATM), Gigabit Ethernet, and serial communications lines. Each physical interface is associated with one device driver (network interface). A single network interface can have more than one IP address.

Note

If a host has multiple interfaces under DHCP (Dynamic Host Configuration Protocol) control and receives a different host name from a DHCP server on each of the DHCP-controlled interfaces, the DHCP client uses the host name it receives on the primary interface to configure the host name for the client. For more information about DHCP, see Chapter 5.

3.3.2 Primary Interface

Although you can have multiple physical interfaces on a single computer, some of the parameters that are configurable by DHCP are interface specific. Examples of interface-specific parameters are the IP address and subnet mask. However, most DHCP configurable parameters are systemwide configurable parameters. Examples of systemwide parameters are the host name and DNS domain name. The TCP/IP Services DHCP client supports controlled configuration of systemwide configurable items by designation of a **primary interface**.

The primary interface is the interface on which the DHCP client uses systemwide parameters received from the DHCP server to configure the system. Systemwide parameters received on an interface that is not designated as primary are not configured on your system by the server. Although only one interface on a system is designated as the primary DHCP interface, the system is not required to have any interface designated as primary.

If a system has multiple interfaces and only one is under DHCP control, you can configure the systemwide parameters manually. DHCP client uses the following rules to resolve conflicts:

- The only-one-primary-interface rule

  This rule solves the potential conflict between two DHCP controlled interfaces on a host getting different systemwide parameter values. To resolve the conflict, you designate one interface to be the primary interface and the parameters that you receive on that interface are the values the DHCP client uses to configure the system. TCP/IP Services does not let you designate two primary interfaces.

- The primary-interface-not-required rule

  This rule solves the problem of DHCP configuring interfaces with an IP address but also keeping manual control of the systemwide parameters. In this case, the DHCP client does not designate the interface as the primary interface, and it ignores any systemwide parameters it receives from a DHCP server.

For details about configuring multiple interfaces, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

3.3.3 Pseudointerfaces

To use extended routing, you can define pseudointerfaces. A **pseudointerface** is a data structure that extends subnet routing using a network interface. Each network interface has one name and at most nine pseudointerface names. Each network interface and pseudointerface has its own IP address, network mask, and broadcast mask.

Like an interface, the name of an internet pseudointerface consists of three alphabetic characters, followed by the pseudointerface unit number in the range of 0 through 255. The first two characters are the same as the two characters in the internet interface name (interface type and interface class). The third character identifies the controller letter that corresponds to the OpenVMS hardware controller. For more information about interface names, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

3.4 Serial Line Connections

A **serial connection** is made between two systems using modems and telephone lines or other serial lines. TCP/IP Services supports serial connections using PPP (Point-to-Point Protocol) and SLIP (Serial Line Internet Protocol). SLIP includes CSLIP (compressed SLIP). You can use any standard OpenVMS terminal device as a PPP or a SLIP line. However, PPP is available for OpenVMS Alpha systems only.

One of the largest applications for IP over serial lines is dialup access. With this type of configuration, your OpenVMS host answers calls and establishes a connection initiated by a user on a client host. The client host can be another OpenVMS system, a UNIX system, or a PC. Alternatively, users on your host can originate the dialup connection to a remote host or terminal server that is running the same protocol. Dedicated serial lines running PPP or SLIP can also be used to connect separate LANs into a single WAN. In such a configuration, the host at each end of the serial connection is always the same; no other hosts are allowed to connect to either serial device.

If your OpenVMS system is part of a large network, you will probably use both PPP and SLIP for your serial connections. As an Internet standard, PPP is often preferred because it ensures interoperability between systems from a wide variety of vendors. PPP provides a way for your OpenVMS Alpha system to establish a dynamic IP network connection over a serial line without additional router or server hardware.

SLIP has been in use for a longer period of time than PPP and is available for most terminal servers and in most PC implementations of TCP/IP. Because SLIP and PPP do not communicate with each other, hosts must use the same protocol in order to communicate. For example, if your terminal server supports only SLIP, remote hosts that connect through this server must also use SLIP. For more information about configuring serial lines, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

For More Information

For more information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

- Configuring and troubleshooting OpenVMS Clusters, including load balancing and failover configurations
- Configuring multiple interfaces and multihomed systems
- Details about pseudointerfaces

- Configuring serial lines

For detailed descriptions of OpenVMS Alpha and VAX similarities and differences, refer to *A Comparison of System Management on OpenVMS AXP and OpenVMS VAX*.

For complete information about supported devices and configurations, refer to the *Guidelines for OpenVMS Cluster Configurations* and the *OpenVMS Cluster Software Software Product Description (SPD)*. For complete information about setting up and using an OpenVMS Cluster environment, refer to the *OpenVMS Cluster Systems* manual.

4

OpenVMS Operating System TCP/IP Features

The OpenVMS operating system contains a number of features that are of specific benefit to the TCP/IP environment. This chapter discusses the following topics related to these features:

- TCP/IP management commands
- Using logical names
- OpenVMS System Dump Analysis (SDA) Tool
- Accessing system messages through operator communication manager (OPCOM) and log files
- Comparison of ODS-5 and ODS-2 file structures
- Print queues (network printers)

Things to Consider

In planning your TCP/IP Services for OpenVMS, consider the following:

- Should I use ODS-5? For which disks?
- Where should I store the log files?
- Which printers in my system are network shareable? How will users access them?
- Which printers in my system are on a serial line?
- Should I configure PATHWORKS shares for printers?

4.1 TCP/IP Management Control Program

The TCP/IP Services Management Control Program is a comprehensive, easy-to-use network management tool that includes more than 100 OpenVMS commands. TCP/IP Services provides this management command interface to configure and modify parameters of components, configure customer-developed services, enable and disable running components, and monitor the running software

To start the management control program, enter the following command:

```
$ TCPIP
TCPIP>
```

At the `TCPIP>` prompt, you can enter commands such as the following:

SHOW SERVICES

SHOW CONFIGURATION

HELP

SET HOST

COPY

DIR

You can also use UNIX management commands to manage some components of TCP/IP Services.

To use UNIX management commands at the DCL prompt, run the following command procedure:

```
$ @SYS$MANAGER:TCPIP$DEFINE_COMMANDS
```

Then enter UNIX commands as you would on a Tru64 UNIX system.

TCP/IP management commands are described fully in the *Compaq TCP/IP Services for OpenVMS Management Command Reference* manual, and in the TCP/IP Services online help.

```
TCPIP> HELP
```

To exit the management control program, enter the following command:

```
TCPIP> EXIT
```

To obtain information about TCP/IP Services, enter the following command at the DCL prompt:

```
$ HELP TCPIP_SERVICES
```

4.2 Defining Logical Names

Logical names allow you to customize component behavior. Logical names can point to directories, database files, and log files.

To define a logical name, enter the following DCL command:

```
$ DEFINE logical-name
```

For more information about these logical names, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

The TCPIP$CONFIG database predefines logical names for various databases. During the menu-driven installation procedure, the software configures either the components you select or all of the TCP/IP Services software components. These defaults are designed to get your system up and running as an internet host with minimal effort. TCPIP$CONFIG creates the database files.

After the initial configuration of a component, you can use logical names to modify the settings of the component-specific parameters. Many logical names are defined as “existence logical names”; that is, they can be either on or off. Any value associated with them is ignored. Others require a value of text string as a definition. Every logical name has a default setting.

For more information about how TCP/IP Services components uses logical names, see relevant chapters in this manual and refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

4.3 OpenVMS System Dump Analysis (SDA) Tool

TCP/IP Services for OpenVMS provides network-specific enhancements to the OpenVMS System Dump Analysis (SDA) tool. For more information about SDA enhancements, refer to DCL online help.

If your system fails, you can run the SDA tool on system reboot to analyze the system crash dump. You can do this by adding command lines to the SYSTARTUP_VMS.COM procedure.

If you are unable to analyze a process dump with the debugger, use the System Dump Analyzer (SDA) utility. Refer to the ANALYZE/CRASH command in online help for more information. For example:

```
$ ANALYZE/CRASH billsystem.dmp

OpenVMS (TM) Alpha system dump analyzer
...analyzing a compressed process dump...

Dump taken on 24-JUL-2002 12:03:40.95
SDA>
```

For details, refer to the *OpenVMS VAX System Dump Analyzer Utility Manual* and the *OpenVMS Alpha System Dump Analyzer Utility Manual*.

4.4 System Messages

You can keep log files of events, changes, and other configuration data in two ways.

- Using **OPCOM** (operator communication manager) — available only if you have system privileges.
- Using log files that most components establish when they are configured.

System messages are saved to either one of these utilities. Both are described in this section.

4.4.1 OPCOM

Any terminal that is connected to the operating system can be established as an operator's terminal if OPCOM (operator communication manager) is running. When an operator who is logged in to an account with OPER privilege enters the REPLY/ENABLE command at the designated terminal, that terminal can be used to respond to user requests and to monitor device status. Operator messages are displayed on the system console terminal unless the terminal is explicitly disabled as an operator's terminal.

To set up a terminal to receive OPCOM messages, enter the following command:

```
$ REPLY/ENABLE
```

4.4.2 Log Files

Event logging can help you manage the TCP/IP Services software. By default, user-defined services do not log events, but event logging is enabled by default for all supplied services. You can configure the product to log events to the operator's console or to a log file, or to both. Every component has a default log file. For more information about log files, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

To set up event logging, enter the following command:

```
TCPIP> SET SERVICE service-name /LOG_OPTIONS=ALL
```

For a list of all the logging options, refer to the SET SERVICE command description in the *Compaq TCP/IP Services for OpenVMS Management Command Reference* manual.

Some product components provide additional event logging capabilities. For more information, see the relevant chapters in this manual.

4.5 ODS-5 and ODS-2 File Structures

OpenVMS implements On-Disk Structure Level 5 (ODS-5). This structure provides the basis for creating and storing files with extended file names. The format was introduced for compatibility with other file systems, such as Windows. You can choose whether or not to convert a volume to ODS-5 on your OpenVMS Alpha systems.

The ODS-5 volume structure provides the following features:

- Long file names
- More legal characters in file names
- Preservation of case in file names

These features are described in detail in the OpenVMS product documentation.

ODS-5 provides enhanced file-sharing capabilities for TCP/IP Services as well as for Advanced Server for OpenVMS (or PATHWORKS for OpenVMS), DCOM, and Java™ applications. Once ODS-5 volumes are enabled, some of the new capabilities can impact certain applications or layered products as well as some areas of system management.

The following sections summarize how the enabling of ODS-5 volumes can impact system management, users, and applications.

4.5.1 Considerations for System Management

RMS access to deep directories and extended file names is available only on ODS-5 volumes mounted on OpenVMS Alpha Version 7.2 systems and higher. Compaq recommends that ODS-5 volumes be enabled only on homogeneous OpenVMS Alpha clusters. If ODS-5 is enabled in a mixed-version or mixed-architecture OpenVMS Cluster, the system manager must follow special procedures and must be aware of the following specific restriction: users must access ODS-5 files and deep directories from OpenVMS Alpha systems only because these capabilities are not supported on earlier versions of the operating system.

4.5.2 Considerations for Users

Users on OpenVMS Alpha systems can take advantage of all Extended File Specifications capabilities on ODS-5 volumes that are mounted on those systems. A user on a mixed-version or mixed-architecture OpenVMS Cluster is subject to some limitations in ODS-5 functionality.

For detailed information about mixed-version or mixed-architecture support, refer to the OpenVMS product documentation.

4.5.3 Considerations for Applications

You can select ODS-5 functionality on a volume-by-volume basis. If ODS-5 volumes are not enabled on your system, all existing applications will continue to function as before. If ODS-5 volumes are enabled, be aware of the following changes:

- OpenVMS file handling and command-line parsing are modified to enable them to work with extended file names on ODS-5 volumes and maintain compatibility with existing applications. The majority of existing, unprivileged applications will work with most extended file names, but some applications might need modifications to work with all extended file names.

- Privileged applications that perform filename parsing and need to access ODS-5 file names or volumes should be analyzed to determine whether they require modification.

On ODS-5 volumes, existing applications and layered products that are coded to documented interfaces, as well as most DCL command procedures, should continue to work without modification.

However, applications that are coded to undocumented interfaces or that include any of the following might need to be modified to function as expected on an ODS-5 volume:

- Internal knowledge of the file system, including knowledge of:

  Data layout on the disk
  Contents of file headers
  Contents of directory files

- File name parsing tailored to a particular on-disk structure.
- Assumptions about the syntax of file specifications, such as the placement of delimiters and legal characters.
- Assumptions about the case of file specifications. Mixed-case and lowercase file specifications are not converted to uppercase. This can affect string-matching operations.

4.6 Network Printers

Resource sharing lets users access network printers as if they were directly connected to the user's local systems. With resource sharing, users can access these resources directly after making the initial connection. This is different from file transfer programs in which files must be transferred completely from the remote system before they can be used.

The printer-sharing components of TCP/IP Services include:

- Line printer/line printer daemon (LPR/LPD), which provides print services to remote and local hosts.
- The TELNET print symbiont (TELNETSYM) provides remote printing services that enables OpenVMS printing features not available with the LPR/LPD print service.
- Serial line connection.
- PC-NFS, which provides authentication and print services for personal computers running PC-NFS.

If a printer is on the network, you must set it up like any OpenVMS printer. For information about setting up OpenVMS printers, refer to the relevant OpenVMS documentation.

4.6.1 Line Printer Daemon (LPD) Service

The Compaq TCP/IP Services for OpenVMS software provides network printing through LPR/LPD. The LPR/LPD service has both a client component (LPR) and a server component (LPD). LPD provides remote printing services for many client hosts, including OpenVMS, UNIX, and Windows NT client hosts. Each print queue is either local or remote. Local print queues handle inbound jobs; remote print queues handle outbound jobs for remote printers.

The print setup utility (TCPIP$LPRSETUP) does the following:

- Updates the related printcap database.
- Creates and starts queues.
- Allows you to add commands to the automatic startup and shutdown command procedures.

To print, users at an OpenVMS client enter the DCL command PRINT.

Users working on UNIX clients typically enter the `lpr` command.

To use the Compaq TCP/IP Services for OpenVMS network printer services, you need the following:

- The remote host name.
- The name of the remote print queue or the local queue name. (LPD accepts both local and remote entries.)
- Compaq PrintServer extensions to use the PRINT/PARAMETERS=*options*=*value* command.
- TCP/IP Services for OpenVMS installed and LPR/LPD enabled on your OpenVMS system.

Both the client component (LPR) and the server component (LPD) are partially included in an OpenVMS queue symbiont. The client is activated when you use one of the following commands

- PRINT—to submit a print job to a remote printer whose queue is managed by the LPD symbiont.
- LPRM—to remove (cancel) a pending print job that was previously spooled.
- LPQ—to view the queue of pending jobs for a remote printer

The LPD server is activated when a remote user submits a print job to a printer that is configured on the OpenVMS server. The LPD server consists of the following two components:

- LPD receiver—a process that handles the incoming request from the remote system over the network. The LPD receiver copies the control file (CF) and data file (DF) that represent the print job to the requested printer's LPD spool directory, and places the control file in the print queue for further processing. The receiver also handles LPQ and LPRM functions from remote clients.
- LPD symbiont—parses the print job's control file, and submits the data files to the designated local printer's print queue.

The same LPD symbiont image is used for both client and server. It acts as the client on queues that are set up for remote printers, and it acts as the server on the local LPD queue. The LPD uses the printcap database to process print requests. The printcap database, located in SYS$SPECIFIC:[TCPIP$LPD]TCPIP$PRINTCAP.DAT, is an ASCII text file that defines the print queues. The printcap entries are similar in syntax to the entries in a UNIX `/etc/printcap` file.

Use the printer setup program LPRSETUP to configure or modify printers. The setup program creates spool directories and log files based on the information you supply. For more information and example setup listings, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

For more information about the following network printing services, refer to the *Compaq TCP/IP Services for OpenVMS User's Guide*:

- Sending print jobs to a printer connected to a remote internet host

- Displaying print queue status
- Canceling print jobs
- Receiving on local (OpenVMS system) print queues print jobs initiated from a user on a UNIX system
- Getting a "finished" notification through SMTP mail

4.6.2 TELNET Print Symbiont

The TELNET print symbiont (TELNETSYM) provides remote printing services that enables OpenVMS printing features not available with the LPR/LPD print service. With TELNETSYM, the local OpenVMS system drives a remote printer as if it were directly connected. This is achieved by attaching a printer to a remote TCP/IP terminal server. The TELNET print symbiont has the following functions:

- Transfers record-oriented data to and from disks and printers.
- Configures printers attached to terminal servers that support TELNET.
- Supports outbound functions (to a remote printer), and offers preformatting to outbound print jobs.

> **Note**
>
> TELNET does not work with terminal servers that use only the local area transport (LAT) protocol. The terminal server must support TCP/IP.

The system that originates the print jobs handles the standard print control functions, such as header-page generation, pagination, queuing, and handling of multiple forms. TELNETSYM extends the OpenVMS print symbiont by redirecting its output to a network channel.

Each TELNETSYM process can control up to 16 print queues. You can control the maximum number of print queues by defining the TCPIP$TELNETSYM_STREAMS logical.

For detailed information about configuring and managing TELNETSYM, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

4.6.3 Serial Line Printer Connections

A serial connection for a remote printer is made between a system and a serial line printer. Compaq TCP/IP Services for OpenVMS supports serial connections using the PPP (Point-to-Point Protocol) and SLIP (Serial Line IP), or CSLIP protocols. You can use any standard OpenVMS terminal device as a PPP or a SLIP line. If the remote system is configured as a gateway to a network, local users can also reach other systems on that network through the serial connection. For more information about serial line configurations, see Chapter 3.

4.6.4 Sharing Network Printers Using PATHWORKS (Advanced Server)

Because everyone on a network uses print services, make sure that network print operations are set up efficiently and cost effectively. The choices that you need to make might include the following:

- Which printers to use
- Which computers to use as print servers
- How to configure shared printers for maximum use

Determine which printers you want to make available to your server community. Some considerations regarding printers include:

- Location

  Select printers that are closest to the physical location of users who require their output.

- Cost of use

  You might want to restrict access to expensive-to-use printers rather than make them available to all network users. Conversely, using one network printer for several groups in a building is less expensive than using separate printers for each group in the building.

- Resolution

  Users who frequently print graphics require printers with higher resolution. Groups who usually print text files can use lower-resolution printers.

A computer can act simultaneously as a print server and a file server. The decision to combine print and file servers might depend on security concerns. Although printers should always be available to their users, you might want to locate a file server in a secure place. Regardless of the size of your network, you most likely will install printers on a few select computers. The only special hardware requirement for print servers is that, if you are using parallel or serial printers, the print servers must have the correct output ports.

Unlike parallel and serial devices, printers with built-in network adapter cards do not have to be adjacent to the print server. Network-interface printers are attached to the network through a built-in adapter card. The location of this type of printer has no effect on printing performance, provided that users and printers are not on opposite sides of a network bridge. A Compaq Advanced Server print server can control a virtually unlimited number of network-interface printers.

The Compaq Advanced Server makes printers available to network users through print shares. In addition, you can use a generic queue when several like printers are available to the user. A generic queue can point to several execution queues and is used to distribute printer work load among several like printers by routing a print job to the first available printer through that printer's execution queue. (If you manage the shared printers from Windows NT, the Advanced Server allows you to set up a printer pool, which is similar in function to an OpenVMS generic queue.)

You can use the Advanced Server ADMINISTER command line interface to add printers (as print queues) and print shares to the Advanced Server and to manage them. Alternatively, beginning with Version 7.3 of the Advanced Server for OpenVMS, you can configure the server to allow management of shared printers from Windows NT using the Windows NT print services. The default is to use the Advanced Server ADMINISTER command line interface.

Each print share points to a single print queue with the same name as the share. Permissions that you assign to the share are applied automatically to the associated print queue. As with any other shared resource, a share can be accessed over the network by users who have the appropriate permissions. Four types of permissions apply to print shares: Print (the default), None (no access), Manage Documents, and Full (full control).

For more information about sharing network printers, refer to the *Compaq Advanced Server for OpenVMS Concepts and Planning Guide*.

4.6.5 PC-NFS

The PC-NFS server provides authentication and print services for PCs running NFS. Users on a PC client can associate the name of the PC printer with an OpenVMS print queue and can print files to the associated queue. However, Compaq recommends that PC clients use other mechanisms for accessing OpenVMS print queues.

To access the NFS server, PC users must have an entry in the proxy database and must have corresponding OpenVMS accounts on the server. For more information about configuring PC-NFS, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

For More Information

For detailed information about configuring and managing TELNETSYM, LPD, and PC-NFS, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

For more information about network printing services, refer to the *Compaq TCP/IP Services for OpenVMS User's Guide*.

For more information about the management control commands and for a list of all the logging options within the SET SERVICE command, refer to the *Compaq TCP/IP Services for OpenVMS Management Command Reference* manual or to online help.

For complete information about ODS-5 features, refer to the OpenVMS documentation set.

For more information about sharing network printers, refer to the *Compaq Advanced Server for OpenVMS Concepts and Planning Guide*.

For more information about preinstallation tasks and the step-by-step installation, refer to the *Compaq TCP/IP Services for OpenVMS Installation and Configuration* guide.

5

Network Server Services

This chapter describes key concepts for the following network server features:

- Network Time Protocol (NTP)
- Routing
- Remote client management (BOOTP/DHCP)
- File Transfer Protocol (FTP)
- SNMP

Things to Consider

In planning your TCP/IP Services for OpenVMS, consider the following:

- Will the system serve as a time server and at what stratum? Where does the authoritative time come from?
- Do I need to remote boot any clients? Which kinds?
- Will the system serve as a router? What kind?
- Which file transfer method should I use: FTP or RCP? What are the security needs, client types, and the purposes of the transfer?
- Will I need to service SNMP programs?

5.1 Network Time Protocol (NTP)

The Network Time Protocol (NTP) synchronizes time and coordinates time distribution throughout a TCP/IP network. TCP/IP Services NTP software is an implementation of the NTP Version 4 specification and maintains compatibility with NTP Versions 1, 2, and 3.

Time synchronization is important in client/server computing. For example, systems that share common databases require coordinated transaction processing and timestamping of instrumental data.

Synchronized timekeeping means that hosts with accurate system timestamps send time quotes to each other. Hosts running NTP can be either a time server or a time client, although they often are both a server and a client. NTP does not attempt to synchronize clocks to each other. Rather, each server attempts to synchronize to Coordinated Universal Time (UTC) using the best available source and the best available transmission paths to that source. NTP expects that the time being distributed from the root of the synchronization subnet is derived from some external source of UTC (for example, a radio clock).

If your network is isolated and you cannot access other NTP servers on the internet, you can designate one of your nodes as the reference clock to which all other hosts will synchronize.

Running an NTP server is optional. If you do set up an NTP server, you must decide whether it will be the authoritative server or whether you will get time from another server.

5.1.1 Time Distributed Through a Hierarchy of Servers

In the NTP environment, time is distributed through a hierarchy of NTP time servers. Each server adopts a stratum that indicates how far away it is operating from an external source of UTC. NTP times are an offset of UTC. Stratum 1 servers have access to an external time source, usually a radio clock. A stratum 2 server is one that is currently obtaining time from a stratum 1 server; a stratum 3 server gets its time from a stratum 2 server, and so on. To avoid long-lived synchronization loops, the number of strata is limited to 15.

Stratum 2 (and higher) hosts might be company or campus servers that obtain time from some number of primary servers and provide time to many local clients. In general:

- Lower-stratum hosts act as time servers.
- Higher-stratum hosts are clients that adjust their time clocks according to the servers.

Internet time servers are usually stratum 1 servers. Other hosts connected to an internet time server have stratum numbers of 2 or higher and may act as time servers for other hosts on the network. Clients usually choose one of the lowest accessible stratum servers from which to synchronize.

5.1.2 How the OpenVMS System Maintains the System Clock

The OpenVMS system clock is maintained as a software timer with a resolution of 100 nanoseconds, updated at 10-millisecond intervals. A clock update is triggered when a register, loaded with a predefined value, has decremented to zero. Upon reaching zero, an interrupt is triggered that reloads the register, and repeats the process.

The smaller the value loaded into this register, the more quickly it reaches zero and triggers an update. In such an instance, the clock runs more quickly. A larger value means more time between updates; therefore, the clock runs more slowly. The amount of time between clock updates is known as a **clock tick**.

5.1.3 How NTP Adjusts System Time

Once NTP has selected a suitable synchronization source, NTP compares the source's time with that of the local clock. If NTP determines that the local clock is running ahead of or behind the synchronization source, NTP uses a general drift mechanism to slow down or speed up the clock as needed. NTP accomplishes this by issuing a series of new clock ticks. For example, if NTP detects that the local clock is drifting ahead by +0.1884338 second, it issues a series of new ticks to reduce the difference between the synchronization source and the local clock.

If the local system time is not reasonably correct, NTP does not set the local clock. For example, if the new time is more than 1000 seconds off in either direction, NTP does not set the clock. In this case, NTP logs the error and shuts down.

NTP maintains a record of the resets it makes along with informational messages in the NTP log file, TCPIP$NTP_RUN.LOG. For more details about event logging and for help interpreting an NTP log file, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

Fore information regarding operating system and daylight saving time issues, refer to the OpenVMS documentation set.

5.1.4 Configuring the Local Host

As the system manager of the local host, you determine which network hosts to use for synchronization and for populating an NTP configuration file with a list of the participating hosts.

You can configure NTP hosts in one or more of the following modes:

- Client/server mode

  This mode indicates that the local host wants to obtain time from the remote server and is willing to supply time to the remote server, if necessary. This mode is appropriate in configurations that involve a number of redundant time servers interconnected through diverse network paths. Most internet time servers use this mode.
- Client mode

  This mode indicates that the local host wants to obtain time from the remote server but it is not willing to provide time to the remote server. Client mode is appropriate for file server and workstation clients that do not provide synchronization to other local clients. In general, hosts with a higher stratum use this mode.
- Broadcast mode

  This mode indicates that the local server will send periodic broadcast messages to a client population at the broadcast/multicast address specified. Normally, this specification applies to the local server that is operating as a sender. To specify broadcast mode, use a broadcast declaration in the configuration file.

For information about additional modes, refer to the TCP/IP Services release notes.

5.1.5 Using the Distributed Time Synchronization Service (DTSS)

Your system might be using the Distributed Time Synchronization Service (DTSS). DTSS is provided as an option with DECnet-Plus and the Distributed Computing Environment (DCE). If you are using DTSS, you must use the procedures supplied with DTSS to set time zone information.

If you are running Version 7.3 or later, you can disable DTSS in favor of running NTP. Define the logical name NET$DISABLE_DTSS to keep DECnet-Plus DECdts from starting.

5.2 Routing

Routing is the act of forwarding datagrams based on information stored in a routing table. Routing allows traffic from your local network to reach its destination elsewhere on the internet. Hosts and gateways on a network use routing protocols to exchange and store routing information.

If the hosts on your network need to communicate with computers on other networks, a route through a gateway must be defined. All hosts and gateways on a network store information about routes in routing tables. With TCP/IP Services, routing tables are maintained on the disk and in dynamic memory.

The TCP/IP Services product provides two types of routing. You can define routes manually (**static** routing), or you can enable routing protocols that exchange information and build routing tables based on the exchanged information (**dynamic** routing).

5.2.1 Static Routing

Because static routing requires manual configuration, it is most useful when the number of gateways is limited and when routes do not change frequently. For information about manually configuring routing, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

5.2.2 Dynamic Routing

Complex environments require a more flexible approach to routing than a static routing table provides. Routing protocols distribute information that reflect changing network conditions and update the routing table accordingly. Routing protocols can switch to a backup route when a primary route becomes unavailable, and can determine the best route to a given destination.

Dynamic routing tables use information that is received by means of routing protocol updates; when routes change, the routing protocol provides information about the changes.

Routing daemons implement a routing policy, that is, a set of rules that specify which routes go into the routing table. A routing daemon writes routing messages to a routing socket, which causes the kernel to add a new route or delete , or modify, an existing route.

The kernel also generates routing messages that can be read by any routing socket when events occur that might be of interest to the process (for example, the interface has gone down or a redirect has been received).

TCP/IP Services implements two routing daemons: the **Routing Daemon (ROUTED)** and the **Gateway Routing Daemon (GATED)**. The following sections provide more information about these daemons.

Routing Daemon (ROUTED)

The ROUTED daemon (pronounced “route-dee”) supports the Routing Information Protocol (RIP). When ROUTED starts, it issues routing update requests and then listens for responses. A system that is configured to supply RIP information responds to the request with an update packet. The update packet contains destination addresses and routing metrics associated with each destination. After receiving a RIP update, the ROUTED uses the information to update its routing table.

For details about how to configure dynamic routing with ROUTED, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

> **Note**
>
> ROUTED supports Routing Information Protocol (RIP) V1 only. ROUTED is considered older technology, and many system administrators are replacing it with GATED.

Gateway Routing Daemon (GATED)

The GATED daemon (pronounced “gate-dee”) supports interior and exterior gateway protocols. It obtains information from several routing protocols and selects the best routes based on that information. You can configure GATED to use one or more of the protocols described in Table 5–1.

Table 5–1: GATED Protocols and RFCs

Protocol	Description	Described in this RFC
Routing Information Protocol (RIP) supports both Versions 1 and 2	RIP is a commonly used interior protocol that selects the route with the lowest metric (hop count) as the best route.	RFCs 1058, 1723
Open Shortest Path First (OSPF) Version 2	Another interior routing protocol, OSPF is a link state protocol (shortest path first). It is better suited than RIP for use in complex networks with many routers.	RFC 1583
Exterior Gateway Protocol (EGP)	EGP exchanges reachability information between autonomous systems. An autonomous system is usually defined as a set of routers under a single administration, using an interior gateway protocol and common metric to route packets. Autonomous systems use exterior routing protocols to route packets to other autonomous systems.	RFC 904
Border Gateway Protocol (BGP)	Like EGP, BGP exchanges reachability information between autonomous systems but supports nonhierarchical topologies. BGP uses path attributes to provide more information about each route. Path attributes can include, for example, administrative preferences based on political, organizational, or security considerations.	RFCs 1163, 1267, 1771
Router Discovery	This protocol is used to inform hosts of the availability of routers that it can send packets to, and to supplement a statically configured default router.	RFC 1256

Note

The list in Table 5–1 is continually updated. For the latest details, refer to the *Compaq TCP/IP Services for OpenVMS Software Product Description* (SPD 46.46.*xx*).

The routing protocols described in Table 5–1 are configured in the GATED configuration file, TCPIP$GATED.CONF. This file contains statements that control tracing options, select routing protocols, manage routing information, and manage independent system routing.

Under GATED, load balancing provides for identical routes based on the reference count and use count (you can observe this through `netstat -r`). GATED chooses from among identical routes the one with the lowest reference count. If there is more than one lowest reference count, it uses the lowest use count.

Although ROUTED allows multiple default routes, it does not monitor interface states. Conversely, GATED monitors interface status changes; however, it does not allow multiple default routes.

For information about configuring dynamic routing with GATED, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

5.3 Remote Client Management (BOOTP/DHCP)

Dynamic Host Configuration Protocol (DHCP), a superset of the Bootstrap Protocol (BOOTP), provides a centralized approach to the configuration and maintenance of IP address space. DHCP allows system managers to configure various clients on a network from a single location.

DHCP allocates temporary or permanent IP addresses from an address pool to client hosts on the network. DHCP can also configure client parameters (such as default gateway parameter), domain name server (DNS) parameters, and subnet masks for each host running a DHCP client.

With DHCP, system managers can centralize TCP/IP network configurations and management tasks involved with network connections. DHCP makes network administration easier by allowing:

- Consistent application of network parameters, such as subnet masks and default routers, to all hosts on a network
- Support for both DHCP and BOOTP clients
- Static (permanent) mapping of hardware addresses to IP addresses
- Dynamic (temporary) mapping of hardware addresses to IP addresses, where the client leases the IP address for a defined length of time

Note

An OpenVMS system running TCP/IP Services can be configured as either a DHCP server or a client, but not as both. Moreover, do not attempt to configure both BOOTP and DHCP; if you do, the configuration generates a warning message.

In addition, the TCP/IP Services implementation of DHCP includes support for DHCP server failover in an OpenVMS Cluster environment. For more information about the OpenVMS Cluster environment, refer to Chapter 3.

As a superset of BOOTP functionality, DHCP offers robust configuration services, including IP addresses, subnet masks, and default gateways.

DHCP is built on the client/server model in the following respects:

- The DHCP server is a host that provides initialization parameters.
- The DHCP client is a host that requests initialization parameters from a DHCP server. A router cannot be a DHCP client.

5.3.1 How DHCP Operates

DHCP consists of two components:

- A mechanism for allocating network addresses to clients

- A set of rules for delivering client-specific configuration parameters from a DHCP server to a client

The server and client communicate to accomplish the following steps:

1. When a DHCP client boots, it broadcasts a DHCP request, asking that any DHCP server on the network provide it with an IP address and configuration parameters.
2. A DHCP server on the network that is authorized to configure this client sends the client a reply that offers an IP address.
3. When the client receives the offer, it can accept it or wait for other offers from other servers on the network.
4. Once the client accepts an offer, it sends an acceptance message to the server.
5. When the server receives the acceptance message, it sends an acknowledgment with the offered IP address and any other configuration parameters that the client requested. (The server only responds to specific client requests; it does not impose any parameters on the client.)
6. If the dynamic address allocation method is used, the IP address offered to the client has a specific lease time that determines how long the IP address is valid.

During the lifetime of the lease, the client repeatedly asks the server to renew. If the client does not renew it, the lease expires.

Once the lease expires, the IP address can be recycled and given to another client. When the client reboots, it can be given the old address, if available, or it can be assigned a new address.

For more information about how DHCP operates, refer to RFC 2131 and RFC 1534.

5.3.2 How DHCP Allocates IP Addresses

With TCP/IP Services, DHCP uses dynamic and static IP address-mapping methods. Table 5–2 describes the allocation methods that service DHCP and BOOTP-only client requests.

Table 5–2: DHCP IP Address Allocation Methods

Method	Applicable Client	Description
Dynamic	DHCP and BOOTP	The DHCP server assigns an IP address from an address pool to a client for a specified amount of time (or until the client explicitly relinquishes the address). Addresses no longer needed by clients can be reused. Use dynamic allocation when: • Clients will be connected to the network only temporarily. • You have a limited pool of IP addresses that must be shared among clients that do not need permanent IP addresses. • IP address are scare, and you need to reclaim retired addresses so you can assign them the new clients being permanently connected to the network. For BOOTP clients, DHCP assigns dynamic IP addresses from the address pool and stores the addresses in the lease database by assigning each lease a time of infinity.
Static	DHCP and BOOTP	The system manager manually assigns an IP address to a client and uses DHCP to pass the assigned address to the client. Use static allocation in an error-prone environment where it is desirable to manage IP address assignment outside of DHCP control.
Finite	BOOTP-only	The DHCP server assigns an IP address from the pool to the BOOTP client and defines a lease time based on certain parameters you define in the SERVER.PCY file. When the lease expires, the DHCP server pings the IP address. If the server receives a reply, it extends the lease and does not offer the address to a new client. If not, the address is free and can be assigned to a new client.

The typical network uses a combination of static and dynamic DHCP addressing. As the local system manager or network administrator, you must decide which IP addressing methods are appropriate for your specific policies and environment.

For detailed information about configuring the different types of addressing for clients on your network, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

5.3.3 Relationship Between DHCP and BOOTP

From the client's perspective, DHCP is an extension of the BOOTP functionality. DHCP allows existing BOOTP clients to operate with DHCP servers without having to change the client's initialization software.

Based on the format of BOOTP messages, the DHCP message format does the following:

- Captures the BOOTP relay agents and eliminates the need for a DHCP server on each physical network segment.
- Allows existing BOOTP clients to operate with DHCP servers.

Messages that include a DHCP message-type option are assumed to have been sent by a DHCP client. Messages without the DHCP message-type option are assumed to have been sent by a BOOTP client.

DHCP improves the BOOTP-only functionality in the following ways:

- DHCP allows the serial reassignment of network addresses to different clients by assigning a network address for a finite lease period.
- DHCP allows clients to acquire all of the IP configuration parameters they need to operate.

Note

BOOTP is considered older technology and many system administrators are replacing it with DHCP.

5.3.4 Client ID

With BOOTP, a client is identified by its unique media access control (MAC) address, which is associated with the network adapter card.

DHCP uses a client identifier (ID) to uniquely identify the client and to associate it with a lease. The client creates the client ID from one of the following types of addresses:

- The MAC address.
- A variation of the MAC address. For example, Windows clients create the client ID by prepending the hardware type to the hardware address.

If the client does not include a client ID in the request, the server uses the client's MAC address.

5.4 File Transfer Services

TCP/IP Services includes the following components enable users to transfer data files between local and remote hosts:

- FTP (File Transfer Protocol), which transfers files between hosts.
- Trivial File Transfer Protocol (TFTP), which downloads and transfers files.
- R commands, which copy files to or from remote hosts.

5.4.1 FTP (File Transfer Protocol)

FTP is a TCP/IP standard, high-level protocol used to transfer files bidirectionally. FTP enables users to access files interactively, list directories on a remote host, delete and rename files on the remote host, and transfer files between hosts.

FTP also provides authentication control, which requires users or clients to correctly enter a login name and password to the server before requesting file transfers. The server can refuse access if login and password combinations are invalid.

FTP allows users who do not have a login name or a password to access certain files on a system using an anonymous login name. This functionality is called **Anonymous FTP** and might include one or more of the following restrictions:

- Limited browsing through the file system. Users can access only the anonymous guest (or home) directory and a public directory. The public directory might contain general bulletin information to which the user has read-only access.
- Access to files from (get) or copying files to (put) the guest directory only.
- Access to files (get) from the public directory only.
- Delete privileges for files in the guest directory that are owned by the anonymous account.

For more information about setting up FTP, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

5.4.2 Trivial FTP (TFTP)

TFTP provides a simple, unsophisticated file transfer service. It is intended for applications that do not need complex interactions between a client and server. TFTP can be hardcoded in read-only memory to execute a network bootstrap program. Once it begins execution, TFTP allows the bootstrap program to use the same underlying protocols that the operating system uses. This makes it possible for one host to boot from a server on another physical network.

TCP/IP Services supports downloading of system images and other types of information for client hosts with TFTP.

TFTP transfers files from a TFTP server to diskless clients or other remote systems. The client initiates the file transfer. If the client sends a read request to the TFTP server, the server attempts to locate this file.

TFTP has the following characteristics:

- TFTP clients are not registered in a database.
- TFTP runs as an unprivileged user in the TCPIP$TFTP account and therefore is restricted to files that the unprivileged user can access.
- TFTP clients are not regulated by the usual OpenVMS user security methods.
- No user name or password is required to use the TFTP service.

For information about how to set up TFTP, refer to the *Compaq TCP/IP Services for OpenVMS Installation and Configuration* manual.

5.4.3 R Commands

The TCP/IP Services software includes client and server implementations of the Berkeley Remote (R) command applications. These applications provide users with the following capabilities:

- RCP – Allows files to be copied between remote hosts.
- RLOGIN — Provides interactive access to remote hosts.
- RSH — Passes a command to a remote host for execution.
- REXEC – Authenticates and executes RCP and other commands.
- RMT/RCD – Provides remote access to magnetic tape and CD-ROM drives.

In addition to password authentication, the R commands use a system based on trusted hosts and users. Trusted users on trusted hosts are allowed to access the local system without providing a password.

Trusted hosts are also called **equivalent hosts** because the software assumes that users who have access to a remote host should be given equivalent access to the local host. The system assumes that user accounts with the same name on both hosts are "owned" by the same user. For example, the user logged in as BETHANY on a trusted system is granted the same access as a user logged in as BETHANY on the local system.

This authentication system requires databases that define the trusted hosts and the trusted users. On UNIX systems, these databases are:

- `/etc/hosts.equiv` — defines the trusted hosts and users for the entire system.
- `rhosts` — defines the trusted hosts and users for an individual user account. This file is located in the user's home directory.

On OpenVMS hosts, the proxy database TCPIP$PROXY.DAT defines trusted hosts and trusted users for the entire system.

Each of these topics is covered in detail in the *Compaq TCP/IP Services for OpenVMS Management* guide.

5.4.4 Differences Between FTP and RCP

Unlike FTP, the RCP protocol provides no method of transferring file type information between the sender and the recipient. It transfers only length, a modified and created timestamp, protection mode, and the byte stream of file data. As a result, RCP is unable to determine the file type of a file it receives.

To revert the file type to a usable format in transfers between OpenVMS systems, if the original file is fixed length or undefined, you can change the attributes on the Stream_LF copy to correspond to the format of the original file. To do so, enter the DCL command SET FILE in the following format:

```
SET FILE/ATTR=(file-attribute[,...])
```

For example, the following command transfers an OpenVMS executable image file (with a fixed record length of 512-bytes, and makes the file executable again.

```
$ SET FILE/ATTR=(rfm:fix, lrl:512) rcp-copied-file.exe
```

You can also use a logical name to change the behavior set by the options.

Although RCP uses secure authentication for security, it has file size limitations that FTP does not have. FTP has no security; passwords are sent in ASCII. RCP sends only the length of the file (in ASCII format). OpenVMS interprets this length as a signed 32-bit integer. Therefore, files transferred using RCP must no more than (2 GB -1) bytes (0x7FFFFFF=2147483647 bytes or roughly 1 byte less than 4194304 RMS 512 byte blocks).

5.5 Simple Network Management Protocol (SNMP)

The Simple Network Management Protocol (SNMP) is network management technology that facilitates the management of a TCP/IP network or internet in a vendor-independent manner. SNMP enables a network administrator to manage the various network components using a set of well-known procedures understood by all components, regardless of the original manufacturers.

Configuring SNMP on your OpenVMS system allows a remote SNMP management client to obtain information about your host and to set system and network parameters.

5.5.1 Configuring SNMP

Systems using SNMP fall into two categories:

- Management consoles (sometimes called clients, network management stations, or directors)
- Agents (sometimes called servers)

The management console is the system that issues a query; the agents run on the system being queried. Queries are sent and received in the form of protocol data units (PDUs) inside SNMP messages, which are carried in user data protocol (UDP) datagrams. You can configure your host so that an SNMP client can obtain information about your host and perform updates on your host's management information base (MIB) data items. For example, you can configure your host to:

- Respond to a client's read requests (Gets) for network information.
- Process client write requests (Sets) on your host's MIB data items.
- Send alert messages (Traps) to a client as a result of events that might need to be monitored (for example, an authentication failure).

Table 5–3 describes the SNMP components and the sample code supplied for custom subagent development.

Table 5–3: SNMP Components

Component	Description
Master agent SNMP Version 2	Process name: TCPIP$SNMP_*n*. Keeps track of managed objects and allows objects to register themselves. Sends information about these objects to remote SNMP management consoles. Also maintains a small set of variables for the MIB II component.
MIB II	Process name: TCPIP$OS_MIBS. Provides information about the TCP/IP protocol stack and other network activity.
Host Resources MIB	Process name: TCPIP$HR_MIB. Provides information about the host system.
MIB converter	Extracts a MIB definition in ASN.1 notation into a MIB definition (.MY) file.
MIB compiler	Compiles a MIB-definition files (for example, CHESS_MIB.MY) into source code templates for use in building subagents.

Table 5–3: SNMP Components (cont.)

SNMP utility programs	Acts as a simple client to obtain a set of values for a MIB and to listen for and send trap messages. For information about using the MIB utility programs, refer to the *Compaq TCP/IP Services for OpenVMS SNMP Programming and Reference* guide.
SNMP subagent example	Implements an example based on the chess game; includes executable and source code.

5.5.2 Ensuring Access to Mounted Data

If the proxy setup between the SNMP server and the NFS server is not correct, the host resources MIB subagent cannot access data that has been mounted.

To ensure access to mounted data, set up a proxy to an anonymous user (for example, to TCPIP$NOBODY) on the NFS server system. For more information about adding proxy entries, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

For More Information

For detailed information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide:

- Event logging
- Help interpreting an NTP log file
- Configuring static routing
- Configuring dynamic routing
- Configuring the different types of addressing for clients on your network
- Configuring FTP
- Using R commands

For more information about NTP OpenVMS issues, visit the OpenVMS FAQ web site at:

```
http://www.openvms.compaq.com/wizard/
```

Search on keyword `NTP`. For information about NTP time settings, refer to the *OpenVMS System Manager's Manual*.

6
Mail Services

Mail Services are an extremely important part of TCP/IP Services. Everyone who uses the network — from administrators, to programmers, to users accesses — this service on a regular basis. This chapter describes Post Office Protocol (POP), SMTP, and IMAP.

Things to Consider

In planning your TCP/IP Services for OpenVMS mail services, consider the following:

- Should I use POP or IMAP for my mail services?
- Can my SMTP clients and servers communicate?
- Will OpenVMS mail headers be translated by the chosen protocol?
- What types of mail clients will I support?
- What types and sizes of files will the mail system encounter?

6.1 Post Office Protocol (POP)

The Compaq TCP/IP Services for OpenVMS Post Office Protocol (POP) server and the SMTP server work together to provide a reliable mail service. POP is a mail repository used primarily by PCs to ensure that mail is accepted even when the PC is turned off. With POP, the PC user need not be concerned with configuring the system as an SMTP server. The user logs on to the client system's mail application, and the POP server forwards any new mail messages from the OpenVMS NEWMAIL folder to the PC. The POP server is an OpenVMS implementation of the Post Office Protocol, Version 3 (RFC 1725) and is based on the Indiana University POP server (IUPOP3).

The POP server is assigned port 110, and all POP client connections are made to this port.

6.1.1 POP Server Process

The POP server is installed with SYSPRV and BYPASS privileges and runs in the TCPIP$POP account, which receives the correct quotas from the TCPIP$CONFIG procedure. The POP server is invoked by the **auxiliary server**.

TCP/IP Services implements the UNIX internet daemon `inetd` function, through the security and event logging of the auxiliary server process. The auxiliary server simplifies application writing and manages overhead by reducing simultaneous server processes on the system. In addition, the auxiliary server does the following:

- Eliminates high overhead resulting from nonstop running of all service processes.
- Uses proxy and service databases to provide system security through authentication of service requests.
- Supports event and error logging.

The POP server uses security features provided in the protocol and in the OpenVMS operating system, as well as additional security measures. These methods provide a secure process that minimizes the possibility of inappropriate access to a user's mail file on the served system.

You can modify the POP server default characteristics, and you can implement new characteristics by defining logical names described in the *Compaq TCP/IP Services for OpenVMS Management* guide.

6.1.2 How to Access Mail Messages from the POP Server

To access mail messages from the POP server, you configure a user name and password or the POP shared secret-password string, into your client mail application.

Your client system opens the TCP connection and attempts to access the server by entering applicable POP commands such as USER (user name) and PASS (password), or APOP (shared secret password). In addition, POP supports the UID command, which some POP clients use, in which the UID (user identification) that POP creates for each mail message is a concatenation of the user name and the date of arrival.

By default, the POP server reads mail from the user's OpenVMS NEWMAIL folder. If you do not instruct the POP server to delete the mail, the server either moves the mail to the MAIL folder (if the logical name TCPIP$POP_USE_MAIL_FOLDER is defined) or keeps it in the NEWMAIL folder (if the logical name TCPIP$POP_LEAVE_IN_NEWMAIL is defined). These logical names are described in the *Compaq TCP/IP Services for OpenVMS Management* guide.

6.1.3 How the POP Server Handles Foreign Message Formats

POP contains minimal support for mail messages that contain foreign formats. Such messages are usually binary and therefore are not transferred to the POP client. Instead, the POP server transfers the message headers, along with a brief message instructing the user to log in and extract the foreign message into a file. Foreign messages are moved into your OpenVMS MAIL folder; the POP server then never deletes.

6.1.4 How the POP Server Authorizes Users

Table 6–1 describes the methods the POP server process uses to authorize user access.

Table 6–1: POP User Authorization Methods

Method	Description
Shared secret password	Most secure POP server access method. Initiated by the client system through the APOP command. Allows a user to become authorized by the POP server without having to send a password over the network. Eliminates a potential path for unauthorized users to obtain a password and break into the system. POP requires a shared secret password from any user who wants to read mail using the APOP authorization method. For information about creating the shared secret password, refer to the *Compaq TCP/IP Services for OpenVMS User's Guide*.
User name and password	Least secure POP server access method. Initiated by the client system through the USER and PASS commands. The POP server authorizes the client to access the desired mailbox based on receipt of a valid user name and password.
OpenVMS SYSUAF settings on user accounts	Access to the POP server is not permitted if: • Either the DISMAIL or DISUSER flags are set for the account. • The account has expired according to the SYSUAF expiration date. • Access has been denied because of an incorrect user name and password.
Ability to disable the USER and PASS commands	Allows the system manager to use the APOP authorization method for all POP clients, the more secure means of user authorization. When you disable the USER and PASS commands (by defining the logical name TCPIP$POP_DISUSERPASS), the POP server responds to the commands with a failure message.

6.1.5 Understanding POP Message Headers

Mail message headers sent by the POP server must conform to the standard specified for SMTP in RFC 822. Because many of the messages received on an OpenVMS system are not in SMTP format (for example, DECnet mail or mail from another message transport system), the POP server builds a new set of headers for each message based on the OpenVMS message headers.

Table 6–2 describes POP headers on forwarded mail messages.

Table 6–2: Forwarded POP Mail Messages Header

POP Message Header	Obtained From
`Date:`	Arrival date of message. Changed to UNIX format.

Table 6–2: Forwarded POP Mail Messages Header (cont.)

`From:`	OpenVMS message `From:` field. Rebuilt to ensure RFC 822 compatibility.
`To:`	OpenVMS Mail `To:` field. Not rebuilt.
`CC:`	OpenVMS Mail `CC:` field. Not rebuilt.
`Subject:`	OpenVMS Mail `Subj:` field. Not rebuilt.
`X-VMS-From:`	OpenVMS Mail `From:` field. Not rebuilt.
`X-POP3-Server:`	Server host name and POP version information. Sent only if logical name TCPIP$POP_SEND_ID_HEADERS is defined.
`X-POP3-ID:`	Message UID. Sent only if logical name TCPIP$POP_SEND_ID_HEADERS is defined.

How POP Rebuilds the OpenVMS Mail From: Field

The most important message header is the `From:` header because it can be used as a destination address if a reply is requested from the POP client. Therefore, the POP server rebuilds the OpenVMS Mail `From:` field in compliance with RFC 822 before sending the header to the POP client.

Table 6–3 describes the types of addresses that can appear in the OpenVMS Mail `From:` field.

Table 6–3: OpenVMS Address Types

Address Type	Address Format
SMTP	SMTP%*legal-address,* where *legal-address* is an address that is compliant with RFC 822 and is commonly in the *user@domain* format.
DECnet	*node*::*username*
User name	*username*
DECnet address within quotation marks	*node*::"*user@host*"
Cluster-forwarding SMTP address	*node*::SMTP%"*user@domain*"

A host name is local if one of the following is true:

- The host name is the same as the substitute domain specified in the SMTP configuration.
- The host name is found in the TCPIP$SMTP_LOCAL_ALIASES.TXT file.

Some POP client systems are confused by the use of personal names when you attempt to reply to a mail message or when the name contains commas or other special characters. If you define the TCPIP$POP_PERSONAL_NAME logical name described in the *Compaq TCP/IP Services for OpenVMS Management* guide, make sure you test the configuration carefully with your POP client systems.

If the logical name TCPIP$POP_IGNORE_MAIL11_HEADERS is defined and the address is an SMTP address, the rebuilt `From:` field is not displayed to the user. In this case, the POP server sends the actual headers from the body of the mail as the mail headers.

6.2 Simple Mail Transfer Protocol (SMTP)

To be reliable, electronic mail systems must be able to cope with situations in which the recipient is temporarily unavailable; for example, if the recipient's host is down or off line. Mail must also be able to handle situations in which some of the recipients on a distribution list are available and some are not.

Simple Mail Transfer Protocol (SMTP) is the TCP/IP standard protocol for transferring electronic mail messages from one system to another. SMTP specifies how systems interact and the format of the mail messages they exchange. The Compaq TCP/IP Services SMTP implementation uses the OpenVMS Mail utility. The OpenVMS Mail utility automatically recognizes an SMTP host address. For example:

```
$ MAIL

MAIL> SEND

To:  jones@widgets.com
```

6.2.1 How SMTP Clients and Servers Communicate

In most implementations, SMTP servers listen at port 25 for client requests. In the TCP/IP Services implementation of SMTP, the SMTP receiver is invoked by the auxiliary server when an inbound TCP/IP connect arrives at port 25 (if the SMTP service is enabled). The auxiliary server runs the command procedure specified in the SMTP service database entry that runs the receiver. The receiver image is SYS$SYSTEM:TCPIP$SMTP_RECEIVER.EXE. The receiver process runs in the TCPIP$SMTP account.

The SMTP symbiont processes all mail on the host. It receives jobs one at a time from the generic SMTP queue and delivers them either locally by means of OpenVMS Mail or remotely by means of SMTP.

After receiving a client request, the SMTP server responds, indicating its status (available or not available). If the server is available, it starts an exchange of control messages with the client to relay mail. (Like FTP, SMTP does not define a message format. SMTP commands are sent as ASCII text, and the SMTP server at the remote host parses the incoming message to extract the command.)

The following steps occur:

1. The auxiliary server listens for requests, starts the SMTP receiver, and accepts the TCP connection.
2. The client identifies itself by sending its fully qualified domain name.
3. The server replies with its own fully qualified domain name.
4. The client sends the full e-mail address of the sender enclosed in angle brackets; if the server is able to accept the mail, it returns a readiness code.
5. The client sends the full mail address (also enclosed in angle brackets) of the message's intended recipient.
6. The client sends the body of the message. A minimum of five control message commands are required to conduct steps 1 through 5.

Table 6–4 describes the control message commands.

Table 6–4: SMTP Client Commands

Command	Description
HELLO	Identifies the originating host to the server host. Use the /DOMAIN qualifier to provide the name of the originating host.
MAIL FROM:<*reverse-path*>	Identifies the address at which undeliverable mail should be returned. Usually is the originating host.
RCPT TO:<*forward-path*>	Address of the intended receiver. If sending mail to multiple recipients, use one RCPT TO command for each recipient.
DATA	Signals the end of the RCPT TO commands and tells the recipient to prepare to receive the message.
QUIT	Signals the end of the RCPT TO commands and tells the recipient to prepare to receive the message.

These commands are described in detail in RFC 821.

The configuration procedure TCPIP$CONFIG sets up the SMTP queues for you. For more information about configuring SMTP, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

6.2.2 Understanding How SMTP Translates OpenVMS Mail Headers

The OpenVMS Mail utility contains up to four headers in a mail message:

- `From:`
- `To:`
- `Subj:`
- `CC:`

SMTP supports a large set of mail headers, including:

- `Resent-Reply-To:`
- `Resent-From:`
- `Reply-To:`
- `Resent-Sender:`
- `Sender:`
- `ReturnPath:`

When it composes an OpenVMS Mail message, SMTP uses the text from the first SMTP header in the list that it finds for the OpenVMS Mail `From:` header.

6.2.3 Understanding SMTP Addresses

SMTP addresses are of the form *userID@domain.name*, where *domain.name* is a domain for which there is a DNS Mail Exchange (MX) record. Mail Exchange records tell SMTP where to route the mail for the domain.

6.3 IMAP

IMAP is the Internet Message Access Protocol. The IMAP Server allows users to access their OpenVMS Mail mailboxes by clients communicating with the IMAP4 protocol as defined in RFC 2060. The supported clients used to access e-mail are PC clients running Microsoft Outlook or Netscape Communicator.

By default, the IMAP Server is assigned port number 143. All IMAP clients connect to this port.

The following sections review the IMAP process and describe how the TCP/IP Services software implements IMAP. If you are not familiar with IMAP, refer to RFC 2060 or introductory IMAP documentation for more information.

6.3.1 IMAP Server Process

The IMAP Server is installed with SYSPRV, BYPASS, DETACH, SYSLCK, SYSNAM, NETMBX, and TMPMBX privileges. It runs in the TCPIP$IMAP account, which receives the correct quotas from the TCPIP$CONFIG procedure. The IMAP Server is invoked by the auxiliary server.

The IMAP Server uses security features provided in the protocol and in the OpenVMS operating system, as well as additional security measures. These methods provide a secure process that minimizes the possibility of inappropriate access to a user's mail file on the served system.

You can modify the IMAP Server default characteristics and implement new characteristics by defining the configuration options described in the TCP/IP Services release notes.

6.3.2 How OpenVMS Mail Folder Names Map to IMAP Mailbox Names

OpenVMS Mail folders are presented to the IMAP client as IMAP mailboxes. All mailboxes are presented to the client in lowercase characters, beginning with an initial capital letter, and with capital letters following each space, at sign (@), opening parenthesis ("("), underscore (_), and hyphen (-).

The OpenVMS NEWMAIL folder requires special treatment . Because the IMAP protocol requires a top-level mailbox called Inbox, the NEWMAIL folder is mapped to Inbox. When the user opens the mailbox called Mail (which maps to file MAIL.MAI), the NEWMAIL folder is not listed so that the user is not confused by seeing the same folder listed twice.

OpenVMS Mail folder names are usually in all uppercase characters but can contain lowercase characters. Any lowercase characters are mapped to an underscore (_) followed by the character's uppercase equivalent. Underscores are mapped to double underscores (_ _), and dollar signs are mapped to double dollar signs ($$).

Table 6–5 shows the effects of folder-name mapping.

Table 6–5: OpenVMS Mail Folder-Name Mapping

OpenVMS Mail Folder Name	IMAP Mailbox Name
HELLO	Hello
Hello	H_e_l_l_o
HELLO-ALL	Hello-All
HELLO_ALL	Hello_ _All
HELLO$ALL	Hello$$All

6.3.3 How the IMAP Server Handles Foreign Message Formats

The IMAP Server determines the correct format for common file types. It does this by checking the beginning of the file for a recognizable file header that matches a set contained in the configuration file TCPIP$IMAP_HOME:TCPIP$IMAP_MAGIC.TXT (analogous to the magic files found on UNIX systems). If a matching file header is found, the server can let the client know the MIME type and subtype of the file.

6.3.4 Understanding IMAP Message Headers

Mail message headers sent by the IMAP Server must conform to the standard specified in RFC 822. Because many of the messages received on an OpenVMS system are not in the RFC 822, or Internet, format (for example, DECnet mail or mail from another message transport system), the IMAP Server builds a new set of headers for each message that is not RFC 822 format and that is based on the OpenVMS message headers.

Table 6–6 describes the headers on mail messages that are forwarded by the IMAP Server.

Table 6–6: IMAP Server Forwarded Message Headers

IMAP Message Header	Obtained From
Date:	Arrival date of message. Changed to Internet format, which shows the day of the week, the date, the time, and the time zone offset from Greenwich Mean Time (GMT). An example of the format is Wed, 30 May 01 16:19:53 +0100.
From:	OpenVMS message `From:` field. Rebuilt to ensure RFC 822 compatibility.
To:	OpenVMS Mail `To:` field. Rebuilt to ensure RFC 822 compatibility.
CC:	OpenVMS Mail `CC:` field. Rebuilt to ensure RFC 822 compatibility.
Subject:	OpenVMS Mail `Subj:` field. Accented characters are RFC 2047 encoded, but the change is not visible to users because IMAP clients reverse the encoding.
X-VMS-From:	OpenVMS Mail `From:` field. Not rebuilt.
X-IMAP4-Server:	Server host name and IMAP version information. Sent only if configuration option Send-ID-Headers is set to True.
X-IMAP4-ID:	Message UID. Sent only if configuration option Send-ID-Headers is set to True.

The IMAP Server sends these message headers to the IMAP Client unless both of the following conditions are true:

- The configuration option Ignore-Mail11-Headers is set to True or is not defined.
- The message text starts with SMTP headers.

6.3.5 How IMAP Rebuilds OpenVMS Mail Address Fields

It is important for the IMAP Server to rebuild the `From:` header, because this header can be used as a destination address if a reply is requested from the IMAP client. The same is true for `To:` and `CC:` headers if the user requests that a reply be sent to other listed recipients. Therefore, the IMAP Server rebuilds these fields in compliance with RFC 822 before sending the header to the IMAP Client.

Table 6–7 describes the different types of addresses that can appear in the OpenVMS Mail address fields.

Table 6–7: Various Address Types

Address Type	Address Format
SMTP	SMTP%`"legal-address"`, where *legal-address* is an address that is compliant with RFC 822 and is commonly in the format *user@domain*.
DECnet	*`node::username`*
User name	*`username`*
DECnet	*address node*::"*user@host*"
Cluster forwarding	*node*::SMTP%*`"user@domain"`* *`SMTP_address`*

A host name is local if one of the following conditions is true:

- The host name is the same as the substitute domain specified in the SMTP configuration.
- The host name is found in the TCPIP$SMTP_LOCAL_ALIASES.TXT file.

Some IMAP client systems are confused by the use of personal names when you attempt to reply to a mail message or when the name contains commas or other special characters. If you define the configuration option Personal-Name described in the *Compaq TCP/IP Services for OpenVMS Management* guide, make sure you test the configuration carefully with your IMAP Client systems before going live to ensure that message replies work successfully.

For More Information

For detailed information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide:

- Defining the system logical names to modify the POP server default characteristics and implement new characteristics
- The logical names TCPIP$POP_USE_MAIL_FOLDER and TCPIP$POP_LEAVE_IN_NEWMAIL for storing POP mail.
- The TCPIP$POP_PERSONAL_NAME logical name.
- SMTP

For more information about the TCP/IP management commands, refer to the *Compaq TCP/IP Services for OpenVMS Management Commands Reference* manual.

For more information about IMAP modifications, commands, and configurations, refer to the TCP/IP Services release notes.

For more information about creating the shared secret string using the APOP authorization method, see the *Compaq TCP/IP Services for OpenVMS User's* guide.

For more information about the SET MX_RECORDS command, see the *Compaq TCP/IP Services for OpenVMS Management Command Reference* guide.

7

Connectivity Services

Compaq TCP/IP Services provides several ways to connect to the network. This chapter discusses the following connectivity methods:

- TELNET
- PPP and SLIP
- NFS
- XDM
- DECnet over TCP/IP

Things to Consider

In planning your TCP/IP Services for OpenVMS configuration, consider the following:

- Should I configure SLIP or PPP?
- Should I configure for DECnet over TCP/IP?
- Do I need to set up NFS?

7.1 TELNET

TELNET is a standard protocol that provides remote terminal connection or login service. TELNET enables users at one site to interact with a remote system at another site, as if the user terminals were connected directly to the remote system. The Compaq TCP/IP Services for OpenVMS product implements TELNET to provide:

- Simultaneous multiple sessions
- IBM 3270 terminal emulation (TN3270)
- Two supported interface formats: DCL style and UNIX style

For more information about managing TELNET, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide. For more information about using TELNET, refer to the *Compaq TCP/IP Services for OpenVMS User's Guide*.

7.2 PPP and SLIP

At the Network Interface layer, standard encapsulation of IP packets are defined for the various hardware types. For example, Ethernet uses the Ethernet frame standard to enclose the data being sent with header fields. Serial line connections use either the Serial Line Internet Protocol (SLIP or CSLIP) or the Point-to-Point Protocol (PPP) (Alpha only).

7.2.1 Assigning an IP Address to Your PPP or SLIP Interface

Every network interface must have its own unique IP address. Interfaces cannot share IP addresses.

If you configure PPP interfaces for multiple remote hosts, the remote hosts can obtain their individual IP addresses from your host when they connect. Similarly, you can configure a PPP interface on your system without knowing your own IP address, and you can obtain the IP address when you connect to a remote system.

Before you establish SLIP communication with a remote host, however, you must obtain the IP address for the host's serial interface and assign IP addresses for each interface you configure on the local host.

When using SLIP, consider placing each serial line in a separate subnetwork. You accomplish this by assigning the same subnet mask for the interfaces at either end of the link.

If you need to use an address in the same subnetwork as your site LAN, use the proxy Address Resolution Protocol (ARP) feature. For more information about ARP, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

7.2.2 Serial Line Internet Protocol (SLIP)

SLIP sends a datagram across the serial line as a series of bytes. Table 7–1 shows how SLIP uses the following characters to determine when a series of bytes should be grouped together.

Table 7–1: SLIP Characters

Character	Function	Hexadecimal Value	Decimal Value
END	Marks the end of the datagram. When the receiving SLIP encounters the END character, SLIP knows that it has a complete datagram.	C0	192
ESC	Indicates the end of the SLIP control characters.	DB	219

SLIP starts by sending an END character. If END is encountered within the datagram as data, SLIP inserts an escape character, sending the two character sequence DB DC instead. If the ESC character appears within the datagram as data, it is replaced with the two-character sequence DB DD. The datagram ends with the END character after the last byte in the packet is transmitted.

There is neither a standard SLIP specification nor a defined maximum packet size for SLIP. The TCP/IP Services implementation of SLIP accepts 1006-byte datagrams and does not send more than 1006 bytes in a datagram.

Compressed SLIP provides header compression that is beneficial for small packets and for low-speed serial links. Header compression improves packet throughput. You can enable CSLIP by using the /COMPRESS qualifier when you enter the SET INTERFACE command.

7.2.3 Point-to-Point Protocol (PPP)

PPP uses a frame format that includes a protocol field. The protocol field identifies the protocol (for example, IP, DECnet, or OSI) to be used for communication between the two hosts. The PPP defines the network frame in a 5-byte header and 3-byte trailer. A PPP frame starts and ends with the control byte 7E hexadecimal (126 decimal). The address and control bytes are constant. The 2-byte protocol field indicates the contents of the PPP frame.

7.3 Network File System (NFS)

The Network File System (NFS) server software lets you set up file systems on your OpenVMS host for export to users on remote NFS client hosts. These files and directories appear to the remote user to be on the remote host even though they physically reside on the local system.

After the NFS server is installed on your computer, you must configure the server to allow network file access.

> **Note**
>
> If your network includes PC clients, you might want to configure PC-NFS.

NFS software was originally developed and used on UNIX machines. For this reason, NFS implementations use UNIX conventions and characteristics. The rules and conventions that apply to UNIX files, file types, file names, file ownership, and user identification also apply to NFS.

Because the TCP/IP Services product runs on OpenVMS, the NFS software must accommodate the differences between UNIX and OpenVMS file systems, for example, by converting file names and mapping file ownership information. You must understand these differences to configure NFS properly on your system, to select the correct file system for the application, and to ensure that your file systems are adequately protected while granting access to users on remote hosts.

7.3.1 Clients and Servers

NFS is a client/server environment that allows computers to share disk space and allows users to work with their files from multiple computers without copying them to their local system. The NFS server can make any of its file systems available to the network by **exporting** the files and directories. Users on authorized client hosts access the files by **mounting** the exported files and directories. The NFS client systems accessing your server might be running UNIX, OpenVMS, or other operating systems.

The NFS client identifies each file system by the name of its mount point on the server. The **mount point** is the name of the device or directory at the top of the file system hierarchy that you create on the server. An NFS device is always named DNFS*n*. The NFS client makes file operation requests by contacting your NFS server. The server then performs the requested operation.

7.3.2 NFS File Systems on OpenVMS

The OpenVMS system includes a hierarchy of devices, directories and files stored on a Files–11 On-Disk Structure Level 2 (ODS-2) or Level 5 (ODS-5) formatted disk. OpenVMS and ODS-2 define a set of rules that govern files within the OpenVMS file system. These rules define the way that files are named and catalogued within directories.

If you are not familiar with OpenVMS file systems, refer to the *OpenVMS System Manager's Manual: Essentials* to learn how to set up and initialize a Files–11 disk.

You can set up and export two different kinds of file systems: a traditional OpenVMS file system or a UNIX style file system built on top of an OpenVMS file system. This UNIX style file system is called a container file system.

Each file system is a multilevel directory hierarchy: on OpenVMS systems, the top level of the directory structure is the master file directory (MFD). The MFD is always named [000000] and contains all the top-level directories and reserved system files. On UNIX systems or with a container file system, the top-level directory is called the **root**.

For information about container file systems and about selecting a file system, refer to Chapter 2.

7.3.3 How the Server Grants Access to Users and Hosts

Once a disk on the OpenVMS system is mapped to a pathname, the MFD or any directory below it can be exported. The server uses the following database files to grant access to users on client hosts:

- The **export database**, TCPIP$EXPORT.DAT, is a collection of entries that store information about the file systems you want to make available to users on client hosts.

  Each entry specifies a directory on the local system and one or more remote hosts that are allowed to mount that directory. A user on a client host can mount any directory at or below the export point, as long as OpenVMS allows access to the directory. Exporting specific directories to specific hosts provides more control than exporting the root of a file system (or the MFD in an OpenVMS system) to all hosts.
- The **proxy database**, TCPIP$PROXY.DAT, is a collection of entries that register the identities of users on client hosts. To access file systems on your local server, remote users must have valid accounts on your OpenVMS host.

  The proxy entries map each user's remote identity to a corresponding identity associated with each user's OpenVMS account. When a user on the client host initiates a file access request, the server checks the proxy database before granting or denying the user access to the file.

These database files are created by TCPIP$CONFIG and can be shared by all OpenVMS Cluster nodes running TCP/IP Services. To control access to these database files, set the OpenVMS file protections accordingly. By default, world access is denied.

For more information about how to create these database files on your server, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

7.3.4 How the Server Maps User Identities

Both OpenVMS and UNIX systems use identification codes as a general method of resource protection and access control. Just as OpenVMS employs user names and UICs for identification, UNIX identifies users with a user name and a user identifier (UID) and one or more group identifiers (GIDs). Both UIDs and UICs identify users on a system.

The proxy database contains entries for each user who accesses a file system on your local server. Each entry contains the OpenVMS user name, the UID/GID pair that identifies the user's account on the client system, and the name of the client host. This file is loaded into dynamic memory when the server starts.

When a user on the OpenVMS client host requests access to a file, the client searches its proxy database for an entry that maps the requester's identity to a corresponding UID/GID pair. (Proxy lookup is performed only on OpenVMS servers; UNIX clients already know the user by its UID/GID pair.) If the client finds a match, it sends a message to the server that contains the following:

- Identity of the requester as a UID/GID pair
- Requested NFS operation and any data associated with the operation

The server searches its proxy database for an entry that corresponds to the requester's UID/GID pair. If the UID maps to an OpenVMS account, the server grants access to the file system according to the privileges set for that account. In the following example, the proxy entry maps a client user with UID=15/GID=15, to the OpenVMS account named ACCOUNT2. Any files owned by user ACCOUNT2 are deemed also to be owned by user UID=15 and GID=15.

```
OpenVMS           Type User_ID      Group_ID          Host_name
User_name

ACCOUNT2                 OND               15               15 *
```

After the OpenVMS identity is resolved, the NFS server uses this acquired identity for all data access, as described in the *Compaq TCP/IP Services for OpenVMS Management* guide.

7.3.5 Granting Access to PC-NFS Clients

TCP/IP Services provides authentication services to PC-NFS clients by means of PC-NFS. As with any NFS client, users must have a valid account on the NFS server host, and user identities must be registered in the proxy database.

Because PC operating systems do not identify users with UID/GID pairs, these pairs must be assigned to users. PC-NFS assigns UID/GID pairs based on information you supply in the proxy database. The following describes this assignment sequence:

1. The PC client sends a request for its UID/GID pair. This request includes the PC's host name with an encoded representation of the user name and password.
2. PC-NFS responds by searching the proxy database and SYSUAF for a matching entry and by checking the password. If a matching entry is located, PC-NFS returns the UID/GID pair to the PC client. The PC stores the UID/GID pair for later NFS requests.
3. If PC-NFS does not find an entry for the PC client in the proxy database, it maps the PC client to the default user TCPIP$NOBODY account. In this case, the client may abort the mount attempt. If the client does complete the mount, restricted access may be granted based on privileges established for the default user account.

For more discussion about the default user, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

7.4 X Display Manager (XDM)

The X Window System, developed at the Massachusetts Institute of Technology, is a network-based graphics window system based on the client/server application model. The X protocol, through which the client and server communicate, runs on UNIX domain sockets, TCP/IP, or DECnet. This means that an X display on one system can display information output from an application running on another system in the network.

An X display is a graphic output device that is known by the X Display Manager (XDM). These devices can include:

- An X terminal

- A workstation that has the X Window System software installed and configured
- A PC running Windows or Windows NT and some X Window System software, such as eXcursion or Exceed

The X Display Manager (XDM) is an X client that manages the login process of a user's X window session. XDM is responsible for displaying a login screen on a display specified by an X server, establishing an X window session, and running scripts that start other X clients. When the user logs out of the X session, XDM is responsible for closing all connections and for resetting the terminal for the next user session.

An earlier version of XDM had limitations that were resolved with the introduction of the XDM Control Protocol (XDMCP). Before XDMCP, XDM used the XSERVERS file to keep track of the X terminals for which it managed the login process. At startup, XDM initialized all X terminals listed in the XSERVERS file. If the X terminal was turned off and then turned on again, XDM had no way of knowing that a new login process should be initiated at the X terminal. To reinitialize the X terminal, the XDM process had to be restarted. XDMCP solves this problem.

With XDMCP, XDM can listen for management requests from X terminals as well as use the XSERVERS file for the X terminals that were not XDMCP compatible. (Most X terminals today are XDMCP compatible.)

The TCP/IP Services implementation of XDM is based on the X11R6.1 release from X Consortium.

7.5 DECnet over TCP/IP

TCP/IP Services software includes the PATHWORKS Internet Protocol (PWIP) driver and the PWIP ancillary control process (PWIP_ACP). The PWIP driver allows OpenVMS systems that are running DECnet over TCP/IP, which is included with the DECnet-Plus for OpenVMS Version 6.0 and later software.

In a multiprotocol networking environment, DECnet-Plus enables OSI and DECnet applications to run over an IP network backbone. The OSI over TCP/IP (using RFC 1006) software enables OSI applications such as FTAM, Virtual Terminal, and X.400 to run over TCP/IP. The DECnet over TCP/IP (using RFC 1859) feature allows traditional DECnet applications to run over TCP/IP. Examples of traditional DECnet applications are `mail`, `cterm`, and `fal`.

With RFC 1006 and RFC 1859, OSI and DECnet applications can accept IP names and addresses. These names and addresses are translated by BIND servers. The DECnet and OSI applications include those supplied by Compaq, third-party applications, and user-written applications.

RFC 1006 is a standard of the Internet community. It defines how to implement ISO 8073 Class 0 on top of TCP. Hosts that implement RFC 1006 are expected to listen on TCP port 102.

DECnet over TCP/IP uses RFC 1859, which defines how to implement ISO 8073, Transport Class 2 Non-Use of Explicit Flow Control on Top of TCP (RFC 1006 Extension). Hosts that implement RFC 1859 are required to listen on well-known TCP port 399.

Decision Point

Use DECnet over TCP/IP if you need to:

□ Link DECnet nodes using TCP/IP.

□ Join two existing DECnet networks without renumbering.

□ Run IP-only traffic in part of the backbone and continue using DECnet applications and user interfaces without extra costs and retraining.

When running DECnet over TCP/IP, you can use an IP host name such as the one in the following example:

```
$ set host remotehst6.acme.com
```

For more information about making connections using DECnet over TCP/IP, see the DECnet-Plus for OpenVMS documentation.

For More Information

For detailed information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide:

- Managing TELNET
- The proxy Address Resolution Protocol (ARP) feature
- Commands to use to edit the container file system files
- Backing up and restoring files or setting up container file systems
- Creating specific database files on your server.
- The acquired identity that NFS server uses for all data access
- Instructions for modifying SYSCONFIG variables to change the default values
- Disabling the default user mapping and setting additional security controls
- Setting ACLs to deny access
- The default user

For more information about using TELNET, refer to the *Compaq TCP/IP Services for OpenVMS User's* Guide.

For more information about access checking, refer to the *OpenVMS Guide to System Security*.

To learn how to set up and initialize a Files–11 disk, refer to the *OpenVMS System Manager's Manual: Essentials*.

8

Domain Name System/BIND (DNS/BIND)

TCP/IP Services for OpenVMS software supports the Berkeley Internet Name Domain (BIND) service, which is a popular implementation of the Domain Name System (DNS). BIND has been ported to many platforms, including UNIX, Windows NT, and OpenVMS.

Before you add BIND servers to your network, you should understand the basic BIND service concepts as they apply to the TCP/IP Services for OpenVMS product. They are described in this chapter in the following topics:

- Overview of the BIND Service
- BIND Service Components
- Domains
- Domain Names
- Zones
- Reverse Domain
- BIND Server Functions
- BIND Server Configuration File
- BIND Server Database Files
- BIND Resolver

Note

BIND Version 9 is supported on Alpha systems only, and future support of BIND Version 8 on VAX systems will be limited. Therefore, if you are using BIND Version 8 on a VAX system, Compaq recommends that you upgrade your BIND server to an Alpha system.

Things to Consider

In planning your TCP/IP Services for OpenVMS configuration, consider the following:

- Should I configure BIND as a resolver only?
- Should I configure BIND as a name server only?
- Should I configure BIND and both a resolver and name server?

8.1 Overview of the BIND Service

DNS has a hierarchical, distributed namespace that makes it easy for people to remember and locate the many hosts located throughout the Internet. Since computers remember and locate the same hosts through a numerical address, computers need a method for converting the host name to a numerical address.

BIND is a lookup service that maps host names to IP addresses and IP addresses to host names in response to queries from other BIND servers and clients in the network. BIND can also provide information on available mail servers and well-known services for a domain.

Based on a client/server model, BIND servers maintain databases of host names, IP addresses, mail records, text records, and other network objects. When client systems require this information, they query the servers.

IP address space allocation is one of the many duties for which ICANN (Internet Corporation for Assigned Names and Numbers), a non-profit corporation, assumes responsibility. ICANN also manages protocol parameter assignment, domain name system management, and root server system management functions, which were previously performed under U.S. Government.

8.2 BIND Service Components

The BIND service contains two parts: the BIND resolver and the BIND server.

- BIND resolver — client software interface that:
  - Formulates queries.
  - Sends queries to BIND servers for answers.
  - Interprets the server's answer.
  - Returns the information to the requesting network application.
- BIND server — server software that responds to client queries by providing:
  - Authoritative or nonauthoritative answers to queries about host names and IP addresses for which the server has an answer.
  - Information about other authoritative servers that can answer queries about host names/IP addresses for which the server does not have an answer.
  - Information about how to get closer to the answer if the server does not have either an answer or information about other authoritative servers
  - Information about mail servers and other network application servers (for example, FTP, TELNET).

8.3 Domains

The Internet name space is based on a hierarchical tree structure. Each node in the tree is referred to as a **domain** or a **subdomain**. A domain is an administrative entity that allows for decentralized management of host names, addresses, and user information. Domains can refer to an administrative point in the name space tree or a specific host. A domain is identified by a domain name and includes the name space at or below the domain name. A subdomain is every domain in the name space below the root domain.

Typically, each domain has a domain administrator responsible for coordinating and managing the domain. The domain administrator registers a second-level or lower domain by interacting with the domain administrator in the next higher-level domain.

The domain administrator's duties include:

- Ensuring reliable service
- Ensuring that the BIND data is current
- Taking prompt action when necessary, for example, if protocols are violated or other serious issues occurs

- Controlling the assignments of the host and domain names

The domain administrator furnishes users with access to names and name-related information both inside and outside the local domain.

8.4 Domain Names

The InterNIC assigns names for all top-level domains as well as domains directly below the top-level domains. Individuals are responsible for assigning lower-level domains and host names.

Each domain has a label. For example, the label for the top-level domain for commercial organizations is `com`. A label is unique within its parent domain.

The concatenation of all the domain labels from the top-level domain to the lowest-level domains is called a **fully qualified domain name**. The labels are listed from right to left and are separated by dots. For example, the domain name for a subdomain within the `com` domain would be `abc.com`; `abc` is the label for the ABC company's subdomain, and `com` is the label for the commercial domain. This structure allows administration and data maintenance to be delegated down the hierarchical tree.

Note

The term **domain name** is sometimes used to refer to a specific domain label. The name of the root domain of the name space is a dot (.).

8.4.1 Types of Domain Names

There are two types of domain names: the fully qualified name and the relative name.

- The fully qualified name represents the complete domain name. This is also known as the absolute or canonical name. For example:

  ```
  boston.cities.compaq.com
  ```

  A domain name that is fully qualified is absolute. You should not append further BIND extensions to the name.
- The relative name represents the starting name (label) of an absolute domain name. Relative names are incomplete but are completed by the BIND service using knowledge of the local domain. Relative host names, such as `boston.cities`, are automatically expanded to the fully qualified domain name when given in a typical command.

8.4.2 Domain Name Format

The format of domain and host labels have the following characteristics::

- Contains characters, digits, or a hyphen.
- Must begin with a character or digit.
- Must not end with a hyphen.
- Has a maximum of 63 characters for each label.
- Has a maximum of 255 characters in a fully qualified domain name.

Although label names can contain up to 63 characters, it is best to choose names that are 12 characters or less because the canonical (fully qualified) domain names are easier to keep track of if they are short. The sum of all the label characters and label lengths cannot exceed 255.

Note

Domain names are not case sensitive. However, the case of entered names is preserved whenever possible.

For example, the fully qualified domain name `euro.sales.compaq.com` is broken down as follows (from right to left):

- The `com` label refers to the commercial top-level domain.
- The `compaq` label refers to the `compaq` domain, a subdomain of the commercial domain.
- The `sales` label refers to the `sales` domain, a subdomain of the compaq domain.
- The `euro` label refers to the host called `euro`, a subdomain of the `sales` domain.

8.5 Zones

For management reasons, a domain can be divided into **zones**, which are discrete, nonoverlapping subsets of the domain. A zone usually represents an administrative or geographic boundary, and authority for the zone may or may not be delegated to another responsible group or person. Each zone starts at a designated level in the domain name tree and extends down to the leaf domains (individual host names) or to a point in the tree where authority has been delegated to another domain.

A common zone is a second-level domain, such as `abc.com`. Many second-level domains divide their zones into smaller zones. For example, a university might divide its domain name space into zones based on departments. A company might divide its domain name space into zones based on branch offices or internal divisions. Authority for the zone is generally delegated to the department or branch office. The department or branch office then has the responsibility for maintaining the zone data.

All the data for the zone is stored on the master server in zone files.

8.5.1 Delegation

When a zone is very large and difficult to manage, authority for a portion of the zone can be delegated to another server; the responsibility for maintaining the zone information is also delegated.

For example, the `edu` zone contains many educational organizations. Each organization is delegated the authority for managing their portion of the `edu` zone, thereby creating subzones. For example, both `rpi.edu` and `uml.edu` are subzones of the `edu` zone and each organization has the responsibility for maintaining the zone information and the master and slave servers for their respective zones.

8.6 Reverse Domains

The Internet has a special domain used for locating gateways and supporting internet address-to-host name lookups. The mapping of internet addresses to domain names is called **reverse translation**. The special domain for reverse translation is the IN-ADDR.ARPA domain.

8.7 BIND Server Functions

If a network consists of relatively few hosts, host name to IP address translations can be accomplished by using a centralized hosts database file.

As soon as a network connects to another network, or when the number of hosts grows large, a more robust method for performing host name/IP address translation is required. In particular, when a network is part of the worldwide internet, no single database can keep track of all addressing information. A considerable number of hosts and network domains are added, changed, and deleted every day.

BIND uses different types of name servers to ensure that all queries are resolved quickly and efficiently:

- Root servers
- Master name servers
- Slave name servers
- Forwarder servers
- Caching-only servers

When a client makes a query, a name server can be in one of three possible states:

- It knows the IP address authoritatively, based on addresses residing in its data files.
- It knows the IP address but not authoritatively, from data cached in its memory from a previous query
- It does not know the address and must refer the query to another server.

The following sections discuss the different types of name servers and their primary responsibilities in the distributed environment of BIND and DNS.

8.7.1 Root Name Servers

Root name servers are the master name servers for the top-level domains of the internet root zone. If the root name server is not the authority for a zone, it knows whom to contact to find out which server is the authority.

If a nonroot server receives a request for a name that is not within its zone, the server starts name resolution at the root zone and accesses the root servers to get the needed information.

The InterNIC determines root servers for the top-level domain, such as A.ROOT_SERVERS.NET, which is a current server name (formerly, `ns.internic.net`). These servers change from time to time. You can obtain the up-to-date list by:

- Copying the named root file maintained at the InterNIC by using FTP anonymous login to `ftp.rs.internic.net` (198.41.0.6). The file is in the domain subdirectory.
- Using the `dig` utility.
- Using the online registration process at the InterNIC web site.

These servers know about all the top-level DNS domains on the Internet. You must know about these servers when you make queries about hosts outside of your local domain. The host names and internet addresses of these machines change periodically. Therefore, check with the InterNIC to obtain changes, and store them in the hints file of the BIND name servers (usually called `ROOT.HINT` on a TCP/IP Services system).

8.7.2 Master Name Server

There are two types of master servers: a master name server and a slave name server (also called a secondary master name server).

The master server is the primary authority for the zone. The master server has complete information about the zone, and it stores the information in its database files. If network information changes, those changes are captured in the master server's database files.

A server can be a master server for more than one zone, acting as the master name server for some zones and a slave name server for others.

You can have more than one master server; however, maintaining two sets of database files requires making the same changes to both sets of files. A more efficient solution is to have one master server and one or more slave servers that obtain their zone information from the master server.

8.7.3 Slave Name Server

A slave name server is an administrative convenience that provides redundancy of information and that shares the load of the master name server. A slave name server receives its authority and zone data from a master name server. Once it is running, a slave name server periodically checks with the master name server for zone changes. If the slave's serial number is less than the master's serial number, the slave requests a zone transfer.

The slave name servers poll the master server at predetermined intervals specified in the zone database files. A time lapse between changing the master server's databases and the slave name servers requesting the update may exist.

8.7.4 Forwarder Servers

Often it is beneficial to limit the traffic to the Internet. The reason might be a slow internet connection or you are being charged by the number of packets.

Funneling DNS Internet queries through one name server can reduce the number of queries going out to the Internet. A name server that performs this function is a **forwarder**. The forwarder handles all off-site queries and in doing so builds up a cache of information; this reduces the number of queries that the forwarder needs to make to satisfy a query.

Forwarder servers have access to the Internet and are able to obtain information regarding other servers that is not currently found in local caches. Because a forwarder server can receive requests from several slave servers, it can acquire a larger local cache than can a slave server. All hosts in the domain have more information available locally because the forwarder servers have a large cache. This means that the server sends fewer queries from that site to root servers on networks outside the internet.

8.7.5 Caching-Only Servers

All servers cache the information they receive for use until the data expires. The length of time a server caches the information is based on a time-to-live (TTL) field attached to the data the server receives.

Caching-only servers have no authority for any zone, and thus do not have complete information for any zone. Their database contains information acquired in the process of finding answers to clients' queries.

8.7.6 Configurations Without Internet Access

You can run the BIND service on a local network that does not have internet access. In this configuration, the servers resolve local queries only. Any request that depends on Internet access goes unresolved.

8.7.7 Zone Transfers

Zone transfers are the process by which slave servers obtain their zone data. When a slave server starts up and periodically thereafter, the server checks whether its data is up to date. It does this by polling a master server to see whether the master server's zone database serial number is greater than the slave's. If so, the slave performs a zone transfer over the network.

An essential point in this polling environment is that whenever a change is made to a master server's zone database file, the zone's serial number must be incremented for the change to propagate to other servers. If the serial number does not change, the slave server does not know it should perform a zone transfer.

Zone Change Notification

In addition to slave servers polling to determine the necessity for a zone transfer, BIND provides a mechanism for a master server to notify slaves of changes to a zone's database.

When a master server determines that a change has been made to a database, it will send a NOTIFY message to all the slave servers for the zone. The slave servers respond with a NOTIFY response to stop any further NOTIFY messages from the master before they query the master server for the **start of authority (SOA)** record of the zone. When the query is answered, the slave checks the serial number in the SOA record and if the serial number changes, the slave transfers the zone. This interrupt feature combined with polling provides a good balance between slow propagation of data because of long refresh times and periods of inconsistent data between authority servers when zone data is updated.

Dynamic Update

DNS Dynamic Update, a BIND feature, provides for zone changes in real time, without having to change a database file and then signal the master server to reload the zone data. Most often, these changes come from other network applications, like DHCP servers, which automatically assign an IP address to a host and then want to register the host name and IP address with BIND.

Dynamic Update provides for:

- Adding and deleting individual resource records
- Deleting a set of resource records with the same name, class, and type
- Deleting all records associated with a given name
- Specifying that prerequisite records exist before adding an address record

Dynamic updates are remembered over system reboots or restart of the BIND server. Whenever the BIND server starts up, it looks for and reads the file where it logged updates (typically, `domain.db_jnl`) and merges the updates into its cache of zone data. While running, the BIND server occasionally writes any pending dynamic updates to the zone database file.

Note

You should not manually edit the zone database file of a zone that is being dynamically updated.

8.8 BIND Server Configuration Files

BIND reads information from an ASCII file called TCPIP$BIND.CONF. On UNIX systems, the file name is `named.conf`. This configuration file consists of statements that specify:

- The location of each BIND database file
- Global configuration options
- Logging options
- Zone definitions
- Information used for authentication

8.9 BIND Server Database Files

Files residing on BIND server systems contain the database of information needed to resolve BIND queries. The following sections describe the four database files used by the server:

- Master zone file
- Reverse zone file
- Loopback interface files
- Hints file

For detailed information about how to create and name these files, refer to the *Compaq TCP/IP Services for OpenVMS Management* manual.

8.9.1 Master Zone File

A master server maintains the master zone file. This file contains:

- Start-of-authority (SOA) records, which specify the domain name for the zone, a serial number, refresh time, retry and other administrative information
- NS records, which specify all the servers for the zone
- Address resource (A) records for each host in the zone
- MX records for mail servers
- CNAME records for specifying alias names for hosts
- Other various resource records

There is one master zone file for each zone for which the server has authority.

8.9.2 Reverse Zone File

For every host with an A record in the master zone file, an IP address must be mapped back to a host name. This is accomplished by using a zone file for a special domain called the IN-ADDR.ARPA domain.

The zone file for this domain contains PTR records that specify the reverse translations (address-to-host name) required for the zone. There is an

IN-ADDR.ARPA zone file for each network represented in the master zone file including the loopback interface.

8.9.3 Loopback Interface Files

The loopback interface files define the zone of the local loopback interface, known as LOCALHOST. There is a master zone file and a reverse zone file for LOCALHOST. The resource record for this file defines LOCALHOST with a network address of 127.0.0.1. TCP/IP Services for OpenVMS configuration procedure creates these two files and calls them LOCALHOST.DB and 127_0_0.DB.

8.9.4 Hints File

The hints file contains information about the authoritative name servers for top-level domains. You can obtain this information from the InterNIC. However, the TCP/IP Services TCPIP$CONFIG procedure creates this file during the configuration procedure.

8.10 BIND Resolver

The BIND resolver is a set of routines that is linked into each network application that needs DNS name resolution services. The resolver formulates one or more queries based on the resolver's configuration and information supplied by network applications; it sends the queries to a server to obtain an answer.

You can configure the following resolver features:

- Define the default domain.
- Specify a domain search list.
- Specify the name servers to query.
- Specify a transport (either UDP or TCP).
- Specify a timeout interval for requests.

The *Compaq TCP/IP Services for OpenVMS Management* guide contains information about how to configure the resolver.

8.10.1 Default Domain

The default domain is the domain in which the client host resides. When resolving a query when just the host name is supplied, the resolver appends the default domain to the host name and then processes the query. This is a convenience for the user. It saves typing a fully qualified domain name.

8.10.2 Search List

The search list is also another convenience for the user. The default search list is derived from the default domain and is applied if the user enters a domain name that is not fully qualified.

8.10.3 Name Servers

You can configure the resolver to query any name server, including the local host, and you can specify a maximum of three name servers. The resolver queries each name server in the order listed until it receives an answer or times out.

For More Information

For detailed information about DNS/BIND, refer to the *Compaq TCP/IP Services for OpenVMS Management* guide.

9

IPv6

Internet Protocol Version 6 (IPv6), as defined in RFC 2460, is the replacement Network layer protocol for the Internet and is designed to replace Internet Protocol Version 4 (IPv4). IPv6 also changes the structure of the Internet architecture. This does not mean that you have to deploy IPv6 all at once across your network; rather, you can make the change in stages because IPv6 and IPv4 were designed to interoperate. This chapter provides guidelines for deployment, deployment scenarios, and checklists for you to consult before you configure a single system or your entire network.

Things to Consider

Before implementing IPv6 into your network, consider the following:

- Is my system part of an IPv6 network?
- What is my internet/intranet scenario?

9.1 Understanding IPv6

The following is a summary of IPv6 features:

- Addressing

  The IPv6 address is 128 bits in length (compared with the 32-bit IPv4 address) and uses a new text representation format. In addition, there are three types of IPv6 addresses: unicast, anycast, and multicast. The unicast address consists of an address prefix and a 64-bit interface identifier. For information about IPv6 addresses, refer to the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual and to RFC 2373.

- Neighbor discovery

  Neighbor discovery is a mechanism by which IPv6 nodes on the same link discover each other’s presence, determine each other’s link-local addresses, find routers, and maintain reachability information about paths to active neighbors and remote destinations. For more information, refer to RFC 2461.

- Stateless address autoconfiguration

  The process by which IPv6 nodes listen for router advertisement packets from routers and learn IPv6 address prefixes. The node creates IPv6 unicast addresses by combining the prefix with a datalink-specific interface identifier that is typically derived from the datalink address of the interface. The OpenVMS operating system performs this process automatically. For more information, refer to RFC 2462.

9.1.1 Mobile IPv6

TCP/IP Services enables an OpenVMS node to operate as a mobile IPv6 correspondent node as defined in the Internet draft “Mobility Support in IPv6” (David B. Johnson and Charles Perkins). For more information about this proposed standard, refer to:

```
http://www.ietf.org/ietf/1id-abstracts.txt
```

Note

This site lists Internet-Drafts documents, all of which are works in progress and subject to change at any time.

The Internet Protocol Version 6 (IPv6) was designed to support mobility through features like its extensible header structure, address autoconfiguration, security (IPsec) and tunneling. mobile IPv6 builds upon these features.

In a mobile IPv6 environment, nodes can have the following roles:

- Mobile node, which is a host or router that can change its point of attachment from one link to another while still being reachable through its home address.
- Correspondent node, which is a peer node with which a mobile node is communicating. The correspondent node (host or router) can be either mobile or stationary.
- Home agent, which is a router on a mobile node's home link with which the mobile node registers its current care-of address. (Currently, OpenVMS cannot operate as a home agent).

A mobile node on its home link has a home address. The subnet prefix of this address is the home network's subnet prefix. The mobile node is always addressable by its home address.

When the mobile node is away from home, on a foreign link, it acquires a care-of address. The subnet prefix of this address is the foreign network's subnet prefix. A mobile node can have multiple care-of addresses, the care-of address registered with the mobile node's home agent is called its primary care-of address.

The association of the mobile node's home address with its care-of address is called a binding. This association has a lifetime. Each node maintains a cache of all bindings.

When the mobile node is on its home link, packets from the correspondent node that are addressed to the mobile node's home address are delivered through standard IP routing mechanisms.

When the mobile node is on a foreign link, it configures a care-of address and registers it with its home agent by sending the home agent a binding update. Packets sent by a correspondent node to the mobile node's home address arrive at its home link. The home agent intercepts the packets, encapsulates them, and tunnels them to the mobile node 's registered care-of address.

After the mobile node receives the tunneled packets, the mobile node assumes that the original sending correspondent node has no binding cache entry for the mobile node care-of address, otherwise the correspondent node sends the packet directly to the mobile node using a routing header.

The mobile node then sends a binding update to the correspondent node. The correspondent node creates a binding between the home address and care-of address.

Packets flow directly between the correspondent node and mobile node. This route optimization eliminates what is commonly known as **triangle routing**. or congestion at the mobile node's home agent and home link. It also reduces the impact of any possible failure of the home agent, the home link, or intervening networks leading to or from the home link, since these nodes and links are not involved in the delivery of most packets to the mobile node.

Away from home, the mobile node sends a home address option to inform the receiver of its home address enabling the receiver to correctly identify the connection to which the packet belongs. When the mobile node returns to its home link, the mobile node sends a binding update to the home agent and to the correspondent node to clear the bindings.

For more information about mobile IPv6, refer to the TCP/IP Services release notes.

9.2 Understanding How Tunnels Work

Tunneling IPv6 packets in IPv4 is a mechanism that allows IPv6 nodes to interoperate with IPv4 hosts and routers. This approach enables the gradual deployment of IPv6 in your network.

OpenVMS systems can have both an IPv4 address and an IPv6 address. An end system with both addresses is considered a v4/v6 host; a router with both addresses is considered a v4/v6 router. A v4/v6 host can use IPv6 to communicate with other v4/v6 hosts on the same communications link. However, when these hosts need to communicate over an IPv4 network, the hosts need to tunnel the IPv6 packets in IPv4 packets in order for the IPv4 routing infrastructure to route the packets to the destination host.

The OpenVMS implementation of tunneling IPv6 packets in IPv4 uses bidirectional configured tunnels to carry IPv6 packets through an IPv4 routing infrastructure; unidirectional tunnels are not supported. This means that a configured tunnel must be created on the nodes at both ends of the tunnel. A bidirectional **configured tunnel** behaves as a virtual point-to-point link. For the remainder of this chapter, the term configured tunnel refers to a bidirectional configured tunnel. A configured tunnel has a source IPv4 address and a destination IPv4 address.

Table 9–1 describes which configured tunnels are possible.

Table 9–1: Tunnel Configurations

Tunnel Configuration	Description	Described in...
Router-to-router tunnel	The v4/v6 routers are connected by an IPv4 infrastructure. For end-to-end communications, this represents only one segment of the total path.	Section 9.3.3
Host-to-router tunnel	The v4/v6 host and v4/v6 router are connected by an IPv4 infrastructure. For end-to-end communications, this represents the first segment of the total path.	Section 9.3.1
Host-to-host tunnel	The v4/v6 hosts are connected by an IPv4 infrastructure. For end-to-end communications, this represents the total path since the tunnel spans the total path.	
Router-to-host tunnel	The v4/v6 router and v4/v6 host are connected by an IPv4 infrastructure. For end-to-end communications, this represents the final segment of the total path.	Section 9.3.2

For more information about tunnels refer to *Compaq TCP/IP Services for OpenVMS Guide to IPv6*.

TCP/IP Services Version 5.3 includes support for a new tunnel IPv6 transition mechanism called 6to4, as defined in RFC 3056.

For more information about the 6to4 mechanism, refer to the TCP/IP Services release notes.

9.3 Developing an Implementation Plan

The following three scenarios, in order of increasing complexity, serve as models for deploying IPv6 in your network:

- Intranet
- Intranet-to-internet
- Intranet-to-internet-to-intranet

The following sections describe each scenario.

9.3.1 Intranet Scenario

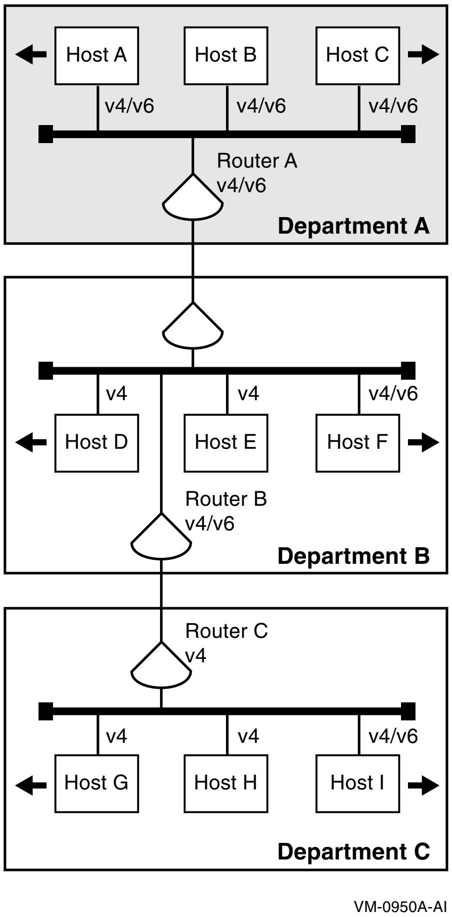
In this scenario, you deploy IPv6 hosts on a small subnet in your network. These hosts communicate with each other using link-local addresses. If you add an IPv6 router to the subnet and advertise an address prefix, each IPv6 host autoconfigures a global IPv6 address and uses that address to communicate with other IPv6 hosts.

As you become more experienced with using IPv6, for the next phase you can add an IPv6 host or hosts on other subnets in your network. Communications between IPv6 hosts on different subnets occur using configured router-to-host tunnels and host-to-router tunnels. The existing IPv4 routing infrastructure is used to get the packets end to end.

The following figures illustrate an intranet scenario in which a corporation has three departments in a local geographic area. Department A has deployed v4/v6 hosts and a v4/v6 router. Departments B and C have deployed only one v4/v6 host each, with a majority of v4 hosts.

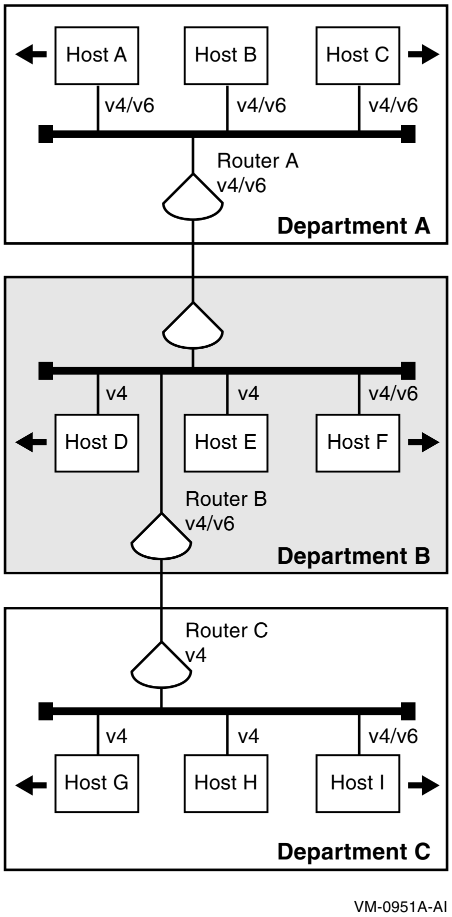
In Figure 9–1, to communicate with host F, native IPv6 traffic is routed from host A to host F via router A.

Figure 9–1: Routing IPv6 Traffic from Host A to Host F



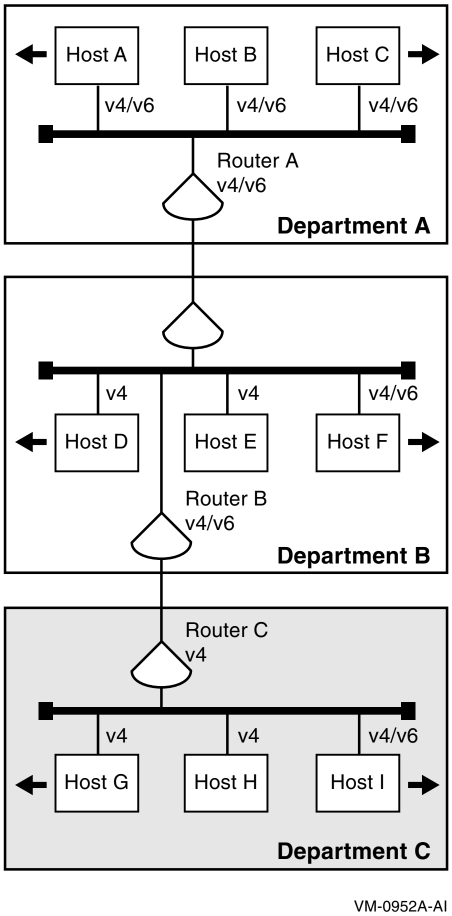
In Figure 9–2, to communicate with host I, host A sends an IPv6 packet to router A. Router A forwards the IPv6 packet to router B. Router B encapsulates the IPv6 packet and sends the IPv4 packet over a router-to-host tunnel to Host I, which decapsulates the IPv4 packet. The IPv4 infrastructure routes the packet to host I. For hosts, the host-to-router tunnel is more efficient because host A, host B, and host C administrators do not need to create individual host-to-host tunnels for each destination host.

Figure 9–2: Routing IPv6 Traffic from Host A to Host I



In Figure 9–3, to communicate with host A, host I encapsulates the IPv6 packet and sends the IPv4 packet over a host-to-router tunnel to router B. From there, router B decapsulates the IPv4 packet and routes the IPv6 packet to host A. For hosts, the host-to-router tunnel is more efficient because the host I administrator does not need to create individual host-to-host tunnels for each destination host.

Figure 9–3: Routing IPv6 Traffic from Host I to Host A



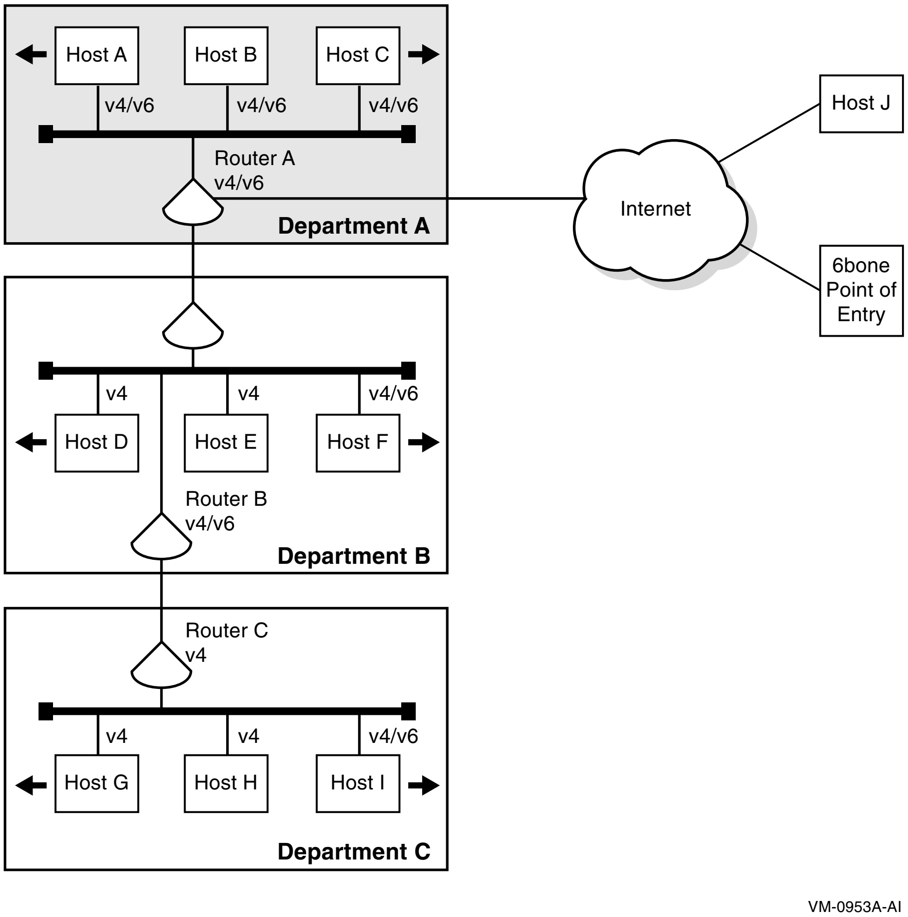
9.3.2 Intranet-to-Internet Scenario

In this scenario, you add a v4/v6 router to your network and use it to communicate with the global Internet. The IPv6 hosts communicate with the v4/v6 router using IPv6. For IPv6 traffic to v4/v6 hosts on the 6bone or the Internet, you configure router-to-host tunnels.

Figure 9–4 illustrates a scenario in which the corporation described in the chapter adds a connection from router A to the Internet. Potential destination nodes are in turn connected to the Internet.

In Figure 9–4, to communicate with host J, host A sends the IPv6 packet to router A. Router A encapsulates the IPv6 packet and sends the IPv4 packet over a router-to-host tunnel to host J, which decapsulates the IPv4 packet.

Figure 9–4: Routing IPv6 Traffic from Host A to Host J



To communicate with the 6bone, host A sends the IPv6 packet to router A. Router A encapsulates the IPv6 packet and sends the IPv4 packet over a router-to-host tunnel to the 6bone point of entry. The point of entry router decapsulates the IPv4 packet and routes the IPv6 packet to its destination.

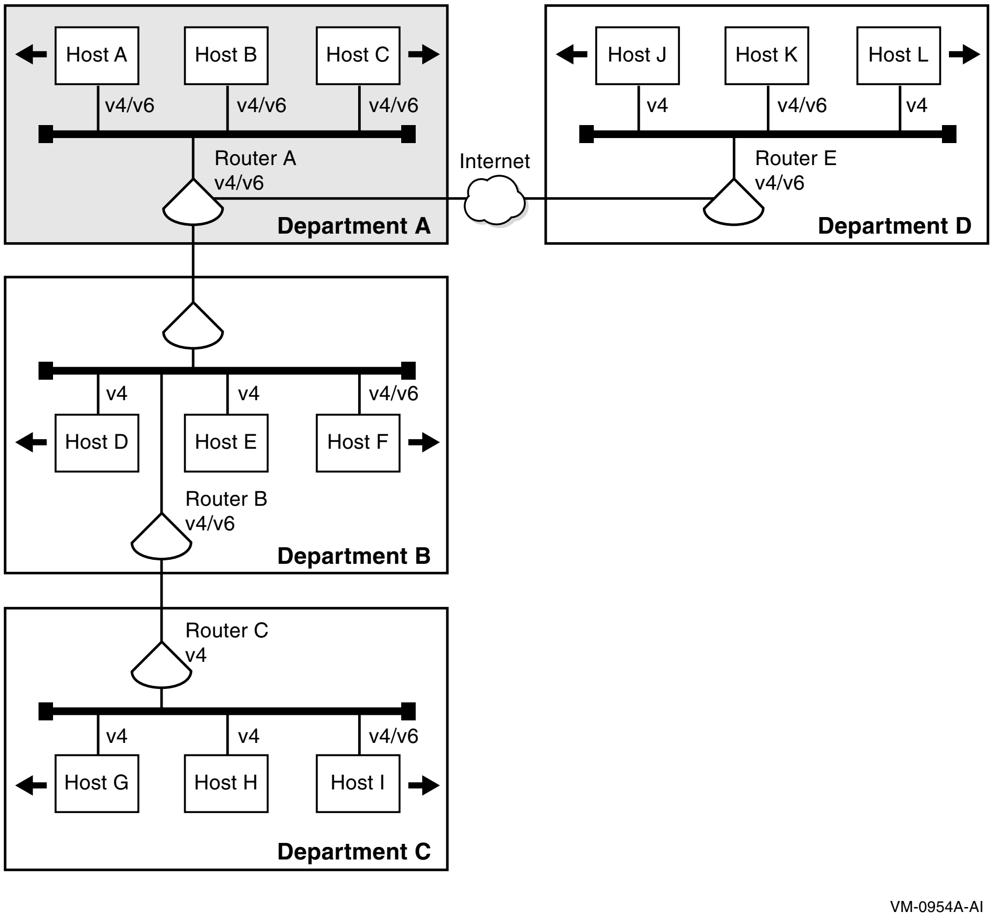
9.3.3 Intranet-to-Internet-to-Intranet Scenario

In this scenario, you add v4/v6 routers on remote subnets and connect the two of them through the Internet to create a virtual private network (VPN). An example of this might be a global corporation with manufacturing in one country and a design center in another country. The IPv6 hosts communicate with the v4/v6 routers using IPv6. For IPv6 traffic between the v4/v6 routers on each subnet, you configure router-to-router tunnels.

Figure 9–5 illustrates a scenario in which the corporation described in the previous sections wants to connect its corporate network with one of its geographically remote departments to create a VPN.

To communicate with host K, host A sends the IPv6 packet to router A. Router A encapsulates the IPv6 packet and sends the IPv4 packet over a router-to-router tunnel to router E, which decapsulates the IPv4 packet and routes the IPv6 packet to host K. For routers, the router-to-router tunnel is more efficient because the router A administrator does not need to create individual router-to-host tunnels for each destination host.

Figure 9–5: Routing IPv6 Traffic from Host A to Host K



9.4 Porting Existing IPv4 Applications

The OpenVMS operating system provides the basic application programming interfaces (APIs) as defined in RFC 2553. You can use the APIs and the AF_INET6 sockets in your existing applications (or in new applications) to communicate with IPv4 nodes today. Your ported applications will continue to communicate with IPv4 nodes and will be ready to communicate with IPv6 nodes. For more information, refer to the *Compaq TCP/IP Services for OpenVMS Sockets API and System Services Programming* manual and the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual.

9.5 Obtaining IPv6 Addresses

IPv6 addresses are now being deployed by the regional registries. To obtain an IPv6 address or block of addresses, contact your Internet Service Provider (ISP).

If you are an Internet Service Provider, contact your upstream registry or one of the registries at the following locations:

- APNIC (Asia-Pacific Network Information Center)
- ARIN (American Registry for Internet Numbers)
- RIPE NCC (Réseau IP Européens)

Because of the need to test various implementation of the IPv6 RFCs, the Internet Engineering Task Force (IETF) has defined a temporary IPv6 address allocation scheme. You can assign the addresses in this scheme to hosts and routers for testing IPv6 on the 6bone. For more information about 6bone address allocation and assignment, refer to the 6bone home page at the following location:

```
http://www.6bone.net
```

After you contract with your ISP for a block of addresses, your deployment of IPv6 in your network begins the process of renumbering of your network. In IPv4, network renumbering was a difficult and time-consuming process. In IPv6, network renumbering is more dynamic. This enables you to renumber your network for any of the following reasons:

- Your enterprise is growing and needs more address space.
- Your network needs are changing.
- Your enterprise wants a global presence.
- You are outgrowing your ISP.

Whatever the reason, when your current ISP contract expires, your right to use the block of IPv6 addresses also expires. Although network renumbering is simplified in IPv6, the following points will help ease the process:

Decision Point

□ Have your routers advertise new network prefixes and deprecate the old prefixes by setting a lifetime.

□ Change DNS servers to advertise node names and the new addresses.

□ Do not hard code addresses in configuration files, because this makes the process more complex and labor intensive.

□ Clear all server caches, as appropriate.

9.6 Installing IPv6-Capable Routers

This process depends on the hardware vendor you have chosen. You will need to define what address prefixes the router will advertise and the interfaces over which to advertise them.

9.7 Configuring Domain Name System/BIND (DNS/BIND) Servers

The OpenVMS operating system supports AAAA lookups over IPv4 (AF_INET) connections only. The resolver and server have not been ported to IPv6, but IPv6 applications can make getaddrinfo and getnameinfo calls to retrieve the AAAA records.

Before you configure a DNS/BIND server to operate in an IPv6 environment, review the following steps:

1. Select a node to function as an IPv6 name server.
2. Dedicate a zone to IPv6 addresses or add IPv6 addresses to your enterprise's current zone. If you want global IPv6 name services, you must delegate a domain under the `ip6.int` domain for the reverse lookup of IPv6 addresses. Do not point different zone names to the same zone database file.
3. See RFC 1886 and RFC 3152 for more information.

If the system is configured as a DNS/BIND server, change the resolver configuration to point to the local node for name lookups.

For more information about configuring Domain Name System, refer to the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual.

9.8 Configuring IPv6 Routers

Before you configure IPv6 routers, consider the following points:

Decision Point

- □ Identify the interfaces over which to run IPv6.
- □ Decide whether you need a configured IPv4 tunnel for communications with other IPv6 nodes or networks. You will need the remote node's IPv4 address (the remote end of the tunnel) and your node's IPv4 address (this end of the tunnel).
- □ Decide whether you want to configure static routes. You might want to configure static routes if one of the following conditions is true:
  - – You want a configured tunnel and you are not advertising an address prefix on the tunnel link.
  - – You want a configured tunnel and the router on the other end of the tunnel is not running the RIPng protocol.
  - – Your system is not running the RIPng protocol.
- □ Identify the interface (LAN, SLIP, or configured tunnel) on which you want to run the RIPng protocol or to advertise an address prefix. If you choose the latter, you must decide on the address prefix to advertise.

For more information, refer to the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual.

9.9 Configuring IPv6 Hosts

Before you configure an IPv6 host, consider the following points:

Decision Point

- □ Identify the interfaces over which to run IPv6.
- □ Decide whether you need a configured IPv4 tunnel for communications with other IPv6 nodes or networks. You will need the remote node's IPv4 address (the remote end of the tunnel) and your node's IPv4 address (this end of the tunnel).
- □ Decide whether you want to configure static routes. You might want to configure static routes if you want a configured tunnel to a router and the router is not advertising itself as a default router on the tunnel link.

For more information, refer to the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual.

For More Information

For detailed information about the following topics, refer to the *Compaq TCP/IP Services for OpenVMS Guide to IPv6* manual:

- IPv6 addresses

- APIs and the AF_INET6 sockets
- Developing applications that use AF_INET6 sockets and client/server code
- Configuring the DNS/BIND server
- Changing the resolver configuration to point to the local node for name lookups
- Configuring IPv6 routers
- Configuring an IPv6 host

For more information about APIs and the AF_INET6 sockets, refer to the *Compaq TCP/IP Services for OpenVMS Sockets API and System Services Programming* guide.

For more information about advanced IPv6 API, refer to the TCP/IP Services release notes.

Glossary

This glossary defines terms that pertain to the features and operation of the Compaq TCP/IP Services for OpenVMS product.

absolute path name

A path name that starts with a slash (/); specifies a file that can be found by starting at the root of the file system and traversing the file tree.

absolute time

A specific date or time of day; specified in the following format: [*dd-mmm-yyyy*] [*:hh:mm:ss:cc*] .

abstract syntax

The description of a data structure that is independent of host structures or codes.

Abstract Syntax Notation One (ASN.1)

The language used by ISO protocols for describing abstract syntax. Most notable use in TCP/IP is for Simple Network Management Protocol (SNMP). The rules of ASN.1 are independent of the encoding techniques used to represent them.

access control information

A character string with login information that validates connect or login at a remote host.

access control list (ACL)

A list that defines the kinds of access to be granted or denied to users.

access rights

A set of privileges that determines what users can do.

ACK

See **acknowledgment**.

acknowledgment (ACK)

A type of message sent to indicate that a block of data arrived at its destination without error. A control bit (acknowledgment flag) in the TCP header indicates that the acknowledgment number field is significant for each segment in a packet.

ACL

See **access control list**.

ACP

See **ancillary control process**.

active port

A port that is bound to a process.

address

A number or group of numbers that uniquely identifies a network node within its own network or internet. *See also* **IP address** and **hardware address**.

address mask

A 32-bit value used to identify which bits in an IP address correspond to the network and subnet portions of the address.

address resolution

The process of relating an IP address to a hardware address, when both refer to the same device, for example, conversion of an IP address into the corresponding Ethernet, Token Ring, or FDDI hardware address. This may require broadcasting on a local network. *See also* **Address Resolution Protocol**.

Address Resolution Protocol (ARP)

The TCP/IP protocol that dynamically binds an IP address to a hardware address such as an Ethernet or FDDI address; limited to physical network systems that support broadcast packets that can be heard by all hosts on a single, physical network. *See also* **proxy ARP**.

addressing

The function that ensures that network systems are correctly identified at all times.

addressing authority

The authority, such as the American National Standards Institute (ANSI), responsible for assigning Network Interface layer addresses within an addressing domain.

addressing domain

A level in a hierarchy of Network Interface layer addresses.

adjacency

A single connection to an adjacent node; collection of state information representing a node in the local node's routing databases.

A relationship formed between selected neighboring routers for the purpose of exchanging routing information. Not every pair of neighboring routers becomes adjacent.

adjacency address

An address that identifies a local subnet access point and a subnet address of an adjacent system.

adjacent nodes

The nodes with direct lines between them; can communicate without an intermediate system. For example, all nodes on an Ethernet LAN are adjacent to each other.

administrative domain

A group of hosts, routers, and networks operated and managed by a single organization. Routing within an administrative domain is based on a consistent technical plan. An administrative domain is viewed from the outside, for purposes of routing, as a cohesive entity, of which the internal structure is unimportant. Information passed by other administrative domains is trusted less than information from one's own administrative domain.

advertisement lifetime

A field in the Router Discovery Protocol router advertisement message that indicates how long advertisement addresses are valid. A lifetime of zero indicates that one or more addresses are no longer valid.

aged packet

A data packet that is discarded because it exceeded the maximum number of hops while being forwarded through the network.

agent

A system that acts on behalf of another system. (1) Client/server model: Part of the system that initiates, prepares, and exchanges information preparation on behalf of a client or server application. (2) Network management: Portion of an entity that responds to management requests and/or preprogrammed trap.

agent access module

The portion of an agent responsible for the agent's end of SNMP.

agent access point

The instance of a connection between a client or director and a server or agent.

agent address

An address that specifies the information needed by a director to establish communications with the agent's management interface.

agent attributes

The attributes maintained by the agent. The attributes do not cross the internal management interface.

aggregate throughput

See **throughput**.

alias

A name, usually easy to remember, that is translated from a different name, usually difficult to remember. Most often used as an optional alternate name for a host. *See also* **host name**.

alias node identifier

An optional node name used by some or all nodes in an OpenVMS Cluster that allows them to be treated as one node.

alternate address notation

The internet address notation that conveys the same information as the common notation, but consists of two parts: network and host.

American National Standards Institute (ANSI)

The organization that coordinates U.S. standards in many areas, including computers and communications.

American Standard Code for Information Interchange (ASCII)

The standard character set that assigns an octal sequence to each letter, number, and selected control characters.

ancillary control process (ACP)

The process that acts as an interface between user software and an I/O driver. The process provides functions supplementary to those performed in the driver, such as file and directory management.

Anonymous FTP

A convention of the File Transfer Protocol that allows a user who does not have explicit authorization to transfer files to and from a host without the need for an account and password. The user usually logs in with a generic user ID and an e-mail address as password.

ANSI

See **American National Standard Institute**.

API

See **Application Programming Interface**.

application

A program that provides functionality for end users of systems.

Application layer

The top-most layer in the Internet architecture model where the user interacts with an application such as Network File Service (NFS), File Transfer Protocol (FTP), and mail.

application process

A part of a distributed application running on a single host.

application programming interface (API)

A standardized set of routines that makes system functions available to programmers.

architecture

The structure of a system, a description of which can be used to recreate the system.

ARP

See **address resolution protocol**.

ASCII

See **American Standard Code for Information Interchange**.

assigned numbers

The numbers officially assigned as part of the Internet standards.

asynchronous transfer mode (ATM)

The method for dynamic allocation of bandwidth using a fixed-size packet (called a cell). Also known as fast packet.

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).

ATM

See **asynchronous transfer mode**.

attribute

The controllable or observable part of an entity; a variable that network managers and applications programmers can manipulate for optimal performance.

attribute group

A named collection of attributes grouped together, such as all information relating to errors.

authentication

Verification of the identity of a person or process attempting to access a system.

authentication server

The software that searches the proxy database for valid user and group identification for remote personal computer users and returns them to PC-NFS.

authority

A name server is said to have authority for a zone. That is, the name server has complete information about a part of a domain space for which the name server is considered to be the authority. A name server may be the authority for one or more zones. Authority for a domain space may be delegated to one or more zones.

authoritative answer

In response to an `nslookup` or a resolver query, an answer is an authoritative answer if a server queries the authority for the zone and returns the answer. A server returns a nonauthoritative answer when the server's answer comes from its own cache.

autonomous confederation

A group of independent computer systems that trust each other regarding routing and reachability information; members believe information provided by other members in preference to information received from systems that are not part of the confederation.

autonomous system (AS)

A collection of networks controlled by one administrative authority. The gateways within this system are expected to trust one another and to share and update routing information among themselves by any mutually agreeable protocol. A core gateway must also be designated to share routing information with other autonomous systems by means of an External Gateway Protocol. *See also* **External Gateway Protocol.**

A set of routers under a single technical administration, using an interior gateway protocol and common metrics to route packets within the AS, and using an exterior gateway protocol to route packets to other ASs. Since this classic definition was developed, it has become common for a single AS to use several interior gateway protocols and sometimes several sets of metrics within an AS.

The use of the term *autonomous system* stresses that even when multiple internal gateway protocols and metrics are used, the administration of an AS appears to other ASs to have a single coherent interior routing plan and presents a consistent picture of what networks are reachable through it. The AS is represented by a number between 1 and 65534, assigned by the Internet Assigned Numbers Authority.

automounting

The process of mounting NFS file systems on an as-needed basis. The NFS file system automatically unmounts after a period of inactivity on the file system. (The default is 5 minutes.) You specify file systems to be automounted in the `automounts` map file.

auxiliary server

The Compaq TCP/IP Services for OpenVMS software that runs as a background process and listens for incoming requests for services. When it receives a request, it runs the appropriate server application; includes `inetd`, security, and logging options.

availability

The proportion of time a specific piece of equipment, system, or network is usable, compared to the total time it is expected to be.

backbone

The primary connectivity mechanism of a hierarchical distributed system. Usually a high-speed high-performance network that links together other networks into an internetwork. All systems with connectivity to an intermediate system on the backbone will connect to each other. This does not prevent systems from setting up private arrangements with each other to bypass the backbone for reasons of cost, performance, or security.

background mounting

In the UNIX environment, the default mount option is to retry remote mount requests in the foreground. If during a boot process, any server listed in `/etc/fstab` is not currently available, the local system will not finish booting until the server becomes available. With background mounting, a remote mount request is executed once in a foreground process. If the mount request fails, the request is retried in a background process. This allows the local system to continue the boot procedure without waiting for the server to become available.

bandwidth

(1) Technically: The difference, in Hertz (Hz), between the highest and lowest frequencies of a transmission channel. (2) Typically: The amount of data that can be sent through a communications circuit.

baseband

A characteristic of any network technology that uses a single carrier frequency and requires all stations attached to the network to participate in every transmission; only one communication channel is provided at a time. *See also* **broadband**.

BBS

See **Bulletin Board System**.

Berkeley Internet Name Domain (BIND)

The implementation of a DNS server developed and distributed by the University of California at Berkeley. Host name and address lookup service for the Internet; implemented in a client/server model. The client software, referred to as the resolver, allows client systems to obtain host names and addresses from servers rather than from locally hosted databases.

Berkeley Software Distribution (BSD)

The 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 Compaq UNIX operating system is based on the BSD version of UNIX.

best-effort delivery

A characteristic of network technologies that will attempt to deliver data but will not try to recover if there is an error such as a line failure. Internet protocols IP and UDP provide best-effort delivery service to application programs.

BG driver

The Compaq TCP/IP Services for OpenVMS implementation of a network device driver. *See also* **device driver**.

BGP

See **Border Gateway Protocol**.

big endian

The format for storage or transmission of binary data in which the most significant bit (or byte) comes first. The reverse convention is called **little endian**.

BIND resolver

A set of library routines compiled into a client application like `telnet` or `ftp` that formulates a query to ask a name server to look up name and address information.

BIND server

The software that responds to queries from BIND resolvers for name and address lookups; can be local or distributed. *See also* **cache server**, **forwarder server**, **primary server**, and **secondary server**.

binding

Defining a remote file system to be a part of the local OpenVMS file system.

bits per second (bps or b/s)

The measure of the rate of data transmission.

block

A contiguous unit of user information grouped together for transmission, such as the user data within a packet, excluding the protocol overhead.

boot file

A database file that BIND servers use to determine their type, the zones for which they have authority, and the location of other BIND database files.

BOOTP

The mnemonic for Bootstrap protocol. The protocol used for booting diskless systems remotely to a network. *See also* **remote boot**.

BOOTP database

A Compaq TCP/IP Services for OpenVMS database with entries for diskless network clients that depend on a boot server to download their operating system images.

Border Gateway Protocol (BGP)

The interautonomous system routing protocol used to exchange network reachability information between autonomous systems. BGP runs over TCP.

One of a class of exterior gateway protocols, described in more detail in the BGP section of UNIX reference page `gated.proto`(4).

bottleneck

A point in the network where traffic is delayed or blocked. Bottlenecks are the limiting factors in network performance.

bound port

An I/O function specifying a port number and IP address for the device socket to bind a port to a process.

bps

See **bits per second**.

bridge

A device that connects two or more physical networks and then stores and forwards complete packets between them. A bridge can usually be made to filter packets (that is, to forward only certain traffic).

broadband

A characteristic of any network that multiplexes multiple, independent network carriers onto a single cable; usually using frequency division multiplexing. Broadband technology allows several networks to coexist on one single cable; traffic from one network does not interfere with traffic from another because the "conversations" happen on different frequencies.

broadcast

A delivery system where a copy of a packet is sent simultaneously to many hosts; can be implemented with hardware (for example, as in Ethernet) or with software (for example, as in Cypress). *See also* **multicast**.

broadcast address

The address that designates all hosts on a physical network. The broadcast address contains a hostid of all ones.

broadcast addressing

A type of multicast addressing in which all nodes receive a message simultaneously.

broadcast circuit

A circuit on which multiple nodes are connected. A message can be transmitted to multiple receivers, and all nodes are adjacent.

broadcast end-node adjacency

An end node connected to the same broadcast circuit as the local node. *See also* **adjacency**.

broadcast router adjacency

An intermediate system (router) connected to the same broadcast circuit as the local node. *See also* **adjacency**.

broadcast mask

A mask used to interpret the IP address as a broadcast address.

broadcast storm

An incorrect packet broadcast on a network that causes most hosts to respond all at once, typically with wrong answers that start the process over again.

brouter

A bridge/router; a device that forwards messages between networks at both network and data link levels.

BSD

See **Berkeley Software Distribution**.

Bulletin Board System (BBS)

A message database where people can log in and leave broadcast messages for others grouped (typically) into topic groups.

buffer

A device or an area of memory used for temporary storage when transmitting data from one device to another. Compensates for a difference in rate of data flow or in time of occurrence of events. Used on routing nodes to temporarily store data that is to be forwarded from one node to another.

buffering level

The number of buffers provided at one time by the network software to handle data. Level can be single or multiple. Single buffering tends to be less efficient than multibuffering but uses less memory on the local system. Multibuffering provides better performance, and a network can send or process several buffers of data in quick succession.

bus

(1) A LAN topology in which all nodes connect to a single transmission medium. All nodes are equal, and all nodes hear all transmissions on the medium. Bus topologies are reliable because failure of a node does not affect the ability of other nodes to transmit and receive. (2) A flat, flexible cable consisting of many transmission lines or wires used to interconnect computer system components to provide communication paths for addresses, data, and control information.

cache

A portion of a computer's RAM reserved to act as a temporary memory for items read from a disk. These items become instantly available to the user.

cache server

A BIND server that has no authority for any zone; acquires information in the process of resolving clients' queries and stores it in its cache. *See also* **BIND server**, **forwarder server**, **primary server**, and **secondary server**.

canonical name

The main or official name for a host; other names for the same host are aliases. In a BIND configuration, you specify the canonical name in a CNAME record of the `named.hosts` file.

category phrase

A BIND configuration logging statement phrase that specifies the different categories for which to log messages. Categories include: `config`, `parser`, `queries`, `lame-servers`, `statistics`, `panic`, `update`, `ncache`, `xfer-in`, `xfer-out`, `db`, `eventlib`, `packet`, `cname`, `security`, `os`, `insist`, `maintenance`, `load`, `response-checks`, and `default`.

centralized management

A form of network management that manages from a single point in the network.

channel

The data path between two or more stations, including the communications control capability of the associated stations.

channel phrase

A BIND configuration logging statement that specifies output methods, format options, and severity levels associated with a category of messages to be logged.

checksum

A computed value based on the contents of a packet. The value is sent with the packet when it is transmitted. The receiving host computes a new value based on the received data. If the originating and receiving values are the same, the receiver has a high degree of confidence that the data was received correctly.

circuit

A logical (virtual) link that provides a communications connection between adjacent nodes.

class name

The name of an entity class. For example, `node` is the global entity class.

client

A computer system or process that requests a service of another computer service or process.

client/server relationship

A model of interaction used in distributed processing products when a client process sends a request and waits for the results from a server process.

clock

The combined hardware interrupt timer and software register that maintain system time. In many systems, the hardware timer sends interrupts to the operating system; at each interrupt, the operating system adds an increment to a software register that contains the time value.

cluster alias

An optional node name and address used by some or all nodes in an OpenVMS Cluster, allowing these nodes to be reachable on the network with the same address.

cluster failover environment

An environment that allows a system in a cluster to take on the responsibilities of a system that crashed or is otherwise unavailable. For example, you can configure a system to become a DHCP server when the primary DHCP server process crashes or when the system that the primary DHCP server is running on becomes unavailable.

collision

The condition in which two data packets are transmitted over a medium at the same time, making both unintelligible.

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.

communications link

The physical medium connecting two systems.

communications server

A special-purpose standalone system dedicated to managing communications activities for other computer systems.

concatenation

The process of joining two or more items together, as when input files are appended to a new output file.

configuration database

The Compaq TCP/IP Services for OpenVMS database with SMTP, SNMP, and TIME specifications.

congestion

The condition in which a network or part of a network is overloaded and has insufficient communication resources for the volume of traffic.

connection

A logical communication path between two processes that are using the TCP protocol. The communication path must exist before data can be sent in either direction. A three-way handshake occurs between the requesting and receiving process to establish a port through which the two processes communicate.

connection-oriented

The model of interconnection that consists of three phases: establish connection, transfer data, and release connection. TCP is a connection-oriented protocol.

connectionless

The model of interconnection in which communication takes place without first establishing a connection. UDP, IP, and IPX are connectionless protocols.

connectivity

The degree to which network nodes are interconnected. Full connectivity means all nodes have links to every other node.

container file

A data file on a Compaq OpenVMS NFS server with a UNIX directory structure and UNIX file attributes for a local, logical UNIX-style file system. Each UNIX regular file is stored as a separate data file. The directory data files in the container file contain the UNIX file names and a pointer to the corresponding OpenVMS Files-11 data file.

container file system

A logical UNIX-style file system that resides on a Files-11 formatted disk and is represented as a set of Files-11 files. *See also* **container file**.

contention

The condition when two or more stations attempt to use the same channel at the same time.

contention control

The scheme of access control used by many networks. Control is distributed among the nodes of the network. Any node wanting to transmit can do so, accessing the network on a first-come, first-served basis. However, it is possible that two nodes are in contention, or start transmitting at the same time, in which case a collision occurs. Each node must then back off and retransmit after waiting a random period of time.

control cluster

A group of small (256-byte) buffers dynamically allocated from nonpaged pool memory; stores information related to device sockets, internal control structures, IP addresses, Internet routes, and Internet packet headers.

Coordinated Universal Time (UTC)

Greenwich Mean Time.

cost

An OSPF (Open Shortest Path First) protocol metric. See metric and OSPF.

counters

The performance and error statistics kept for an entity by network management, such as lines and nodes.

CRC

See **cyclic redundancy check**.

cyclic redundancy check (CRC)

An error detection scheme whereby a number is derived from a set of data before it is transmitted. Once transmitted, the receiving node recalculates the number and compares it to the value originally transmitted. If the numbers are different, some type of transmission error has occurred.

daemon

A process that executes in the background waiting for some event to occur.

data cluster

A group of large (1792-byte) buffers that store data in the system space; transmit and receive operations service user processes by moving data to and from data clusters.

Data Encryption Key (DEK)

Used for encryption of message text and (with certain choices among a set of alternative algorithms) for computation of message integrity check (MIC) quantities.

Data Encryption Standard (DES)

A type of encryption scheme approved by the U.S. National Bureau of Standards.

data link

A logical connection between two systems on the same circuit on which data integrity is maintained.

Data Link layer

The layer in a network model that handles communication between physical hosts.

data octet

See **octet**.

data overrun

The data blocks received that arrived too quickly to be processed by the receiver and were, therefore, lost.

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

The result of fragmenting a datagram. Fragments carry 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 reassembly time

The time allowed for reassembly of a fragmented datagram.

datagram service

The mode of delivery for a datagram which is delivered in such a way that the receiver can determine the boundaries of the datagram as it was entered by the source.

DCE

See **Distributed Computing Environment**.

DCL

See **DIGITAL Command Language**.

decision

The routing process that determines the path, or route, along which a data packet travels to reach its destination; forwards packets on the lowest-cost path even if that one does not have the fewest hops. The path that the data takes through the network is transparent to users.

decoding

The process by which the transfer syntax representation of a data value is transformed into the local representation of that value.

dedicated serial connection

A permanent connection between two hosts using an RS232 serial port. SLIP or PPP can be used for TCP/IP communication between the two hosts.

default route

The route used to direct any data addressed to network host addresses for which no explicit route is specified.

delay

A HELLO metric. Valid values are from 0 to 30000, inclusive. The value of 30000 is the maximum metric and means unreachable. *See* **metric** and **HELLO**.

delete access

The access right that grants users the ability to remove data from the domain.

DEK

See **data encryption key**.

DES

See **Data Encryption Standard**.

designated router

In OSPF, a designated router is a multiaccess network that has at least two attached routers. The designated router generates a link state advertisement for the multiaccess network and assists in running the protocol. The designated router is elected by the HELLO protocol.

destination address

The IP address that specifies where a datagram is to be sent; contains the network and host identifiers.

Any network or host.

destination port

A 2-octet value in the TCP and UDP header field that identifies the destination upper-level protocol for a packet's data.

device driver

The software associated with each physical device; serves as the interface between the operating system and the device controller.

device socket

The extension of the pseudodevice, used for communications; consists of the Internet pseudodevice and the socket. *See also* **pseudodevice**.

DHCP

See **Dynamic Host Configuration Protocol.**

dialogue

The sequence of message exchanges between open systems that represents a single association and the set of underlying connections.

dialup

A temporary (as opposed to dedicated) network connection established through a telephone line with a modem.

dialup provider

A host that responds to incoming PPP connection requests. A PPP server.

DIGITAL Command Language (DCL)

The command interface of the OpenVMS operating system.

Compaq TCP/IP Services for OpenVMS

The Compaq software product implemented on OpenVMS as an ancillary control process (ACP) and a network device driver (BG driver) with executive-level components and user applications that use TCP/IP protocols.

distance

An EGP metric. See metric and EGP. Valid values are from 0 to 255 inclusive.

Distributed Computing Environment (DCE)

An architecture of standard programming interfaces, conventions, and server functions (for example, naming, distributed file system, remote procedure call) for transparently distributing applications across networks of heterogeneous computers.

distributed database

A collection of several different data repositories that look like a single database to the user. The Domain Name System (DNS) is a distributed database.

distributed management

A form of network management in which network managers and management software are dispersed across many systems.

distributed processing

The technology that enables the distribution throughout the network of computing power and storage facilities to user work areas, such as offices, laboratories, or machines on factory floors.

distributed system

A collection of computer systems, tied together by communications networks for the purpose of sharing resources; end users do not need to be aware of the physical location of the shared resources.

DNS

See **Domain Name System**.

domain

An organizational unit with administrative responsibility for naming networks or hosts. An internet domain name consists of a sequence of names (labels) separated by periods (dots); for example, `tundra.mpk.ca.us`.

domain name

The name used to refer to a fully qualified domain or subdomain. For example, in `cat.food.iams.com`, `food.iams.com`, `iams.com`, and `.com` are all domain names. Each name specifies a different domain level.

Domain Name System (DNS)

A distributed database system that allows TCP/IP applications to resolve a host name into a correct IP address.

dot address

See **dotted-decimal notation**.

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.

downline loading

Transferring a copy of a system image from a load host to a target. Some systems, such as DEC WANrouter systems and Compaq DECserver terminal servers, automatically request a downline load of their image upon startup and reboot. One of the functions of a TFTP server.

drift

The change in a clock's time rate over a specified period.

A measure, in Hertz per second, of how quickly the skew of a clock is changing. *See also* **skew**.

dynamic adaptive routing

The automatic rerouting of traffic based on a sensing and analysis of current actual network conditions; not including cases of routing decisions taken on predefined information.

Dynamic Host Configuration Protocol

A superset of the BOOTP protocol that enables the automatic assignment of IP addresses to clients on networks from a pool of addresses. The IP address assignment and configuration occurs automatically whenever appropriate client systems (workstations and portable computers) attach to a network. The TCP/IP Services for OpenVMS implementation of DHCP is based on the JOIN product by Competitive Automation.

dynamic routing

A type of routing where a host or router talks to adjacent routers to learn what networks each router is connected to. Subsequently, the kernel's routing tables are updated when the router learns new information. There are many routing protocols including Interior Gateway Protocols (RIP, OSPF) and Exterior Gateway Protocols (EGP and BGP).

ephemeral port number

A port number temporarily assigned to a client process for the duration of a session. When the client process terminates, the port number can be assigned to another process. The port number is usually from 1024 to 5000.

EGP

See **Exterior Gateway Protocol**.

elective protocol

The classification in Internet standards for optional protocols.

electronic mail

The service whereby a computer user can exchange messages with other computer users (or groups of users) by means of a communications network; one of the most popular uses of the Internet.

e-mail

See **electronic mail**.

encapsulation

A technique used by layered protocols in which a layer adds header information to the protocol data unit (PDU) from the layer below. As an example, in Internet terminology, a packet would contain a header from the physical layer, followed by a header from the Network layer (IP), followed by a header from the Transport layer (TCP), followed by the application protocol data.

encryption

A process of encoding information so the meaning of its content is no longer immediately obvious to anyone who obtains a copy of it.

end node

See **end system**.

end system

A nonrouting system; can receive data packets addressed to it and send data packets to other systems on the same subnet but cannot relay, route, or forward data packets to other systems.

entity

An individual, manageable piece of a network; has attributes that describe it, a name that identifies it, and an interface that supports management operations.

entity class

A collection of entities that share the same properties and have the same parent entity; each member of the class has a unique identifier within the class. Entity classes have class names.

entity group

An architecturally defined collection of entities. The entities in the group must have a common top entity and must all be of the same class.

entity hierarchy

A logical hierarchical tree structures of manageable entities in which child entities are below their parent entities. Children can be accessed only through their parents' agent.

entity identifier

An attribute that specifically identifies an entity. *See also* **attribute group**.

entity name

A label associated with some entities used to identify or locate them for management purposes.

entity type

The subgrouping of an entity that determines its relationship to other entities.

Ethernet

A baseband network medium. Commonly used to connect a local area network.

event

A measurable network-specific or system-specific occurrence for which a logging component maintains a record.

experimental protocol

The classification in Internet standards for protocols that are developed as part of an ongoing research project not related to an operational service offering; not intended for operational use.

export database

The Compaq TCP/IP Services for OpenVMS database with directory names that can be mounted from remote NFS clients.

exported file

A file in an exported directory or a subdirectory of an exported directory. *See also* **exporting**.

exported file systems

A file system that can be accessed by a remote system using the Network File System. The local system imports the remote file system. Both the remote and local system must be configured to grant and receive access to the file system.

exporting

Identifying a directory on an NFS server that can be remotely mounted by NFS clients.

Extended File Specifications

A feature of OpenVMS Alpha Version 7.2 that allows the use of Windows-style file specifications. This feature provides greater flexibility for OpenVMS Alpha systems to store, manage and access files that have names similar to those in a Windows 95 or Windows NT environment.

extended LAN

Multiple LANs connected with data link relays or bridges.

Exterior Gateway Protocol (EGP)

The protocol that distributes routing information to the gateways that interconnect networks.

A class of routing protocols used to exchange routing information within an autonomous system.

One of a class of exterior gateway protocols.

FDDI

See **Fiber Distributed Data Interface**.

fetch/store operation

The operation of 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)

The high-speed (100 mb/s) networking standard based on fiber optics, established by the American National Standards Institute (ANSI); uses 1300 nanometer light wavelength. FDDI networks are limited to approximately 200 km in length, with repeaters every 2 km or less.

file

A uniquely named collection of information with shared managerial and structural properties.

file attribute

The characteristic of a file, such as its size or creation date. The values of some file attributes may change during the lifetime of a file.

file data

The information that is stored within a file and comprises its contents (as opposed to its attributes).

file designation

System-specific information that identifies a file on its storage system.

file server

The host whose principal purpose is to store files and provide network access to them.

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.

File Transfer Protocol (FTP)

The protocol and software that permit a user on one host to access and transfer files to and from another host over a network. *See also* **Trivial File Transfer Protocol**.

Files-11 On-Disk Level 2 or Level 5 (ODS-2 or ODS-5)

The set of rules that govern the organization of the OpenVMS file system, external to the files themselves.

FINGER utility

The utility that provides information about users on local and remote systems.

flow control

(1) The function of a receiving entity to limit the amount or rate of data that is sent by a transmitting entity. (2) The 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 and allow communicating layers to match their data transfer and receive rates. Simplistic schemes, like ICMP source quench, simply ask the sender to cease transmission until congestion ends. More complex schemes vary the transmission rate continuously.

forwarder server

The name server that processes recursive requests that a slave server cannot resolve locally; has access to the Internet. *See also* **BIND server**, **cache server**, **primary server**, **secondary server**, and **slave server**.

forwarding information base

The table that GATED uses internally to store routing information it learns from routing protocols is a routing table; also known as a routing information base, or RIB. The routing table is used to collect and store routes from various protocols.

forwarding table

The table in the kernel that controls the forwarding of packets is a forwarding table, also known as a forwarding information base, or FIB.

FQDN

See **fully qualified domain name**.

fragment

A piece of a packet that results from a router dividing 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, where the data in the fragment occurred in the original datagram. IP software at the receiving end must reassemble the fragments. *See also* **maximum transmission unit**.

fragmentation

The IP process of breaking up packets into smaller packets for transmission; allows a packet originating in a network that allows a large packet size to traverse a network that limits packets to a smaller size. The destination host reassembles the fragments. *See also* **maximum transmission unit**.

frame

A Data Link layer packet that contains the header and trailer information required by the physical medium.

FTP

See **File Transfer Protocol**.

full-duplex circuit

A circuit designed for transmission in both directions at the same time. *Contrast with* **half-duplex circuit**.

full-duplex transmission

Data transmission in both directions at the same time. *Contrast with* **half-duplex transmission**.

fully qualified domain name (FQDN)

The full site name of a system, such as `scryber.enet.dec.com` rather than just the host name of `scryber`.

function code

A parameter in a $QIO system service call that defines the specific function of that $QIO.

GATED

A routing daemon that can be configured to route one or more of the following protocols: RIP, BGP, EGP, and OSPF.

gateway

A communications device or program that passes data between networks having similar functions but dissimilar implementations. The term *router* is now used in place of the original definition of *gateway*.

An intermediate destination by which packets are delivered to their ultimate destination.

A host address of another router that is directly reachable through an attached network. As with any host address it may be specified symbolically.

gateway client

Another term for an access system.

Gateway Routing Daemon

See **GATED**.

GID

See **group identification**.

gigabit

One billion bits.

gigabyte

One billion bytes.

group identification (GID)

The identification code for a group of UNIX users.

half-duplex circuit

A circuit designed for transmission in either direction, but only one direction at one time. *Contrast with* **full-duplex circuit**.

half-duplex transmission

Data transmission in either direction, but only one direction at a time. *Contrast with* **full-duplex transmission**.

handshaking sequence

The exchange of connection information between two communicating entities; takes place to enable the successful completion of a connection. Used, for example, in establishing a TCP connection between client and server applications.

hardware address

The address that identifies the connection device between the network controller of a host and the network cable. *See also* **address**.

hard link

A mechanism that allows you to assign more than one name to a file. Both the new name and the file being linked must be in the same file system. *See also* **link**.

header

The portion of a packet that precedes the actual data and contains control information such as source and destination address and error checking.

header compression

A technique used by PPP and SLIP protocols to reduce the number of bytes per frame when sending packets over a slow serial link. The use of header compression is negotiated between the client and servers processes to reduce the size of the IP and TCP headers.

heterogeneous network

A network consisting of different network protocols or different operating system software, such as OpenVMS and UNIX.

hierarchical routing

Routing based on domains. Interdomain routers are responsible only for getting data to the right domain and intradomain routers take responsibility for routing within the domain.

hop count

The number of connections between two hosts, based on the number of different routers needed to traverse the distance between the two hosts.

hop

A term used in routing. Number of hosts separating a source and final destination (including the final destination) on a network.

host

A computer system that acts as a source or destination of network messages sometimes called **node**.

The IP address of any system. Usually specified in dotted-decimal notation. There are four values in the range from 0 to 255 inclusive, separated by dots (.). For example, 132.236.199.63 or 10.0.0.51. It can also be specified as an 8-digit hexadecimal string preceded by 0x. For example, 0x0a000043. In addition, if the options `noresolv` statement is not specified, this can be a symbolic host name. For example, `gated.cornell.edu` or `nic.ddn.mil`. The numeric forms are preferred over the symbolic form.

host address

See **host number**.

hosts database

The Compaq TCP/IP Services for OpenVMS database that is created by default; allows users to use host names; contains host names, IP addresses of the hosts, and any alias names for the hosts.

host name

The name given to a network host. *See also* **fully qualified domain name** and **alias**.

host number

The part of an IP address that identifies which host on the network is being addressed.

Host-to-Host Communication layer

Also called Transport layer. The second-highest level in the Internet architecture model; 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.

IAB

See **Internet Architecture Board**.

IBM TN3270

The TELNET options that allow TELNET users to connect to hosts that support 3270 terminals.

ICMP

See **Internet Control Message Protocol**.

IETF

Internet Engineering Task Force. A large international community of network designers, operators, vendors and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. Membership is open to everyone. See the `http://www.ietf.org/` web site for more information.

IGP

See **Interior Gateway Protocol**.

IMAP

The Internet Message Access Protocol. IMAP enables clients to access email messages and folders from an IMAP server and synchronize them locally. This enables a client to organize email messages and folders without continuous access to the server.

`inetd`

A UNIX internet daemon. A server process listens for client requests for specific services. When `inetd` receives a request for a service, it starts the appropriate server process.

initial sequence number

The first sequence number used for sending or receiving on a connection.

inode

A UNIX file structure used to address a file block. There is a unique inode allocated for each active file with a name made up of a device/i-number pair.

interface

The boundary between two parts of a system across which communication can occur; may be defined through hardware or software.

The host address of an attached network interface. This is the address of a broadcast, nbma, or loopback interface, and the remote address of a point-to-point interface. As with any host address, it can be specified symbolically.

The connection between a router and one of its attached networks. A physical interface can be specified by a single IP address, domain name, or interface name (unless the network is an unnumbered point-to-point network). Multiple levels of reference in the configuration language allow the identification of interfaces by using wildcard, interface type name, or delete word address. Be careful using interface names because future versions might allow more than one address per interface. Dynamic interfaces can be added or deleted and indicated as up or down as well as changes to address, netmask, and metric parameters.

Interior Gateway Protocol (IGP)

The protocol used to propagate network reachability and routing information within an autonomous system; RIP is among the most popular.

One of a class of routing protocols used to exchange routing information within an autonomous system.

interface list

A list of one or more interface names, including wildcard names (names without a number) and names that may specify more than one interface or address, or the token all-for-all interfaces.

intermediate system

An OSI system that performs Internet layer forwarding. A routing system receives data packets from a system on one subnet and passes them on to a system on another subnet; it receives data packets from a source end system, or from the previous intermediate system on the route, and passes them on to the destination end system, or to the next intermediate system on the route.

internet

A shortened form of internetwork; a network of networks; interconnected TCP/IP networks that function as one large virtual network. Differs from the Internet by their lack of connectivity with the global Internet.

Internet

The worldwide network of networks and gateways that use the TCP/IP protocol suite and function as one virtual network; provides universal connectivity and three levels of network services: unreliable, connectionless packet delivery; reliable, full-duplex stream delivery; and application level services such as electronic mail that build on the first two. The Internet connects many universities, government research labs, military installations, and private businesses.

Internet architecture

A four-layered communications model that consists of the following: Application layer, Transport layer, Internet layer, and Network Interface layer.

Internet Architecture Board (IAB)

The technical body that oversees the development of the Internet suite of protocols (TCP/IP). It has a research task force and an engineering task force, each responsible for investigating a particular area.

Internet Autonomous System

A system that consists of a set of gateways, each of which can reach any other gateway in the same system using paths by means of gateways only in that system. The gateways of a system cooperatively maintain a routing database using an interior gateway protocol.

Internet Control Message Protocol (ICMP)

An extension to the Internet Protocol; used by gateways to communicate with the network software in hosts.

Internet header length

An IP header field that indicates the number of 32-bit words making up the Internet header.

Internet layer

The layer in the TCP/IP network model where data is transferred between hosts across networks. Also referred to as Network Interface layer.

Internet number

See **IP address**.

Internet Protocol (IP)

A connectionless, best-effort, packet-switching protocol that resides in the Internet layer and has two major functions: internet addressing and fragmentation of messages.

Internetwork

A collection of many different computing systems which communicate with each other. The computing systems can include different hardware architectures, operating systems, and network technologies.

interoperability

The ability of software and hardware on multiple machines from multiple vendors to communicate meaningfully.

InterNIC Registration Services

The Internet Network Information Center; organization that provides the Internet community with registration, directory, database, and information services.

I/O status block (IOSB)

A data structure associated with the $QIO system service. The IOSB holds information about how the I/O request completes.

IP

See **Internet Protocol**.

IP address

An address that identifies 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.

IP datagram

The basic 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 indicating whether the datagram can be (or has been) fragmented. An IP datagram is to the Internet what a hardware packet is to a physical network. *See also* **datagram**.

IP forwarding

A configurable kernel option that controls whether a host forwards IP datagrams. Generally, hosts do not forward IP datagrams.

IP trailer protocol

A protocol in which the protocol header follows the data.

KA9Q

A popular implementation of TCP/IP and associated protocols for amateur radio systems.

Kbps

See **kilobits per second**.

kernel

The software that provides the standard API for application programs. Generally speaking, the kernel embodies the policy and structure of an operating system. In a narrower sense, the kernel provides a programmatic interface to any hardware resources available. In a UNIX system, the kernel is a program that contains the device drivers, the memory management routines, the scheduler, and system calls; always running while the system is operating.

kilobits per second (Kbps or Kb/s)

The measure of data transmission rate.

LAN

See **local area network**.

layer

(1) The 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) A software protocol levels that make up network architectures; each layer performs certain functions for the layers above and below it.

limited use protocol

A classification in Internet standards for protocols that are intended for use in limited circumstances; possibly because of their experimental state, specialized nature, limited functionality, or historic state.

line printer daemon (LPR/LPD)

The Compaq TCP/IP Services for OpenVMS remote printing services for UNIX and OpenVMS client hosts.

line speed

The maximum rate at which data can be reliably transmitted over a line; varies with the capability of the modem or hardware device that performs the transmitting.

link

A directory entry referring to a file; one file can have several links to it.

little endian

The format for storage or transmission of binary data in which the least significant byte comes first. The reverse convention is called **big endian**.

load broker

A TCP/IP Services component that provides configurable, calculated methods for distributing BIND services among systems in a cluster.

local address

The address of a host within a subnet.

The host address of an attached interface. This is the address of a broadcast or loopback interface, and the local address of a point-to-point interface. As with any host address, it can be specified symbolically.

local area network (LAN)

A self-contained group of computers and communications devices (such as modems, routers, servers, and repeaters) that offers a high-speed, reliable communications channel. LANs span a limited distance such as a building or group of buildings, but can be connected to wide area networks (WANs) with gateways. *Contrast with* **wide area network (WAN)**.

local data

Any data stored locally by a system.

local network

A network directly attached to a host or gateway.

local node

A node at which the user is located.

local subnet

A subnet directly attached to a host or gateway.

lock manager

An NFS component that allows an NFS client to lock portions of files that reside on an NFS server.

logical connectivity

The ability of nodes to communicate.

logical link

A temporary connection between processes on source and destination nodes (or between two processes on the same node).

Logical Link Control

The upper portion of the Data Link layer that presents a uniform interface to the user of the data link service, usually the Internet layer.

loop node

A local node that is associated with a particular address and is treated as if it were a remote node. All traffic to the loop node is sent over the associated address; used for loopback testing.

loopback

A program that sends packets to a remote host on the Internet and looks for replies; works by means of the echoing facility provided by the ICMP protocol and is a way to determine if an Internet host is reachable from your host. *See also* **packet internet groper**.

LPR/LPD

See **remote line printing** or **line printer daemon**.

mail bridge

A mail gateway that forwards electronic mail between two or more networks while ensuring that the messages it forwards meet certain administrative criteria; specialized form of mail gateway that enforces an administrative policy with regard to what mail it forwards.

Mail Exchange record (MX record)

The Domain Name System resource record type indicating which host can handle mail for a particular domain or host.

Mail exchange (MX)

The Compaq TCP/IP Services for OpenVMS implementation of a mail exchanger that allows hosts in a local network to forward mail to systems that might not be directly connected to the local network.

mail exploder

The part of an electronic mail delivery system that allows a message to be delivered to a list of addressees. Users send messages to one address (e.g., hacks@somehost.edu) and the mail exploder handles delivery to the individual mailboxes.

mail gateway

A host that connects two or more electronic mail systems (especially dissimilar mail systems on two different networks) and transfers messages between them.

mail path

A series of hosts used to direct electronic mail from one user to another.

Management Information Base (MIB)

A database used by the Simple Network Management Protocol (SNMP) to check network statistics and configurations. An SNMP management station can query a MIB or set it in an SNMP agent (for example, router). Standard, minimal MIBs have been defined (MIB I, MIB II), and vendors often have custom entries. In theory, any SNMP manager can talk to any SNMP agent with a properly defined MIB.

Management Information Base II (MIB-II)

Data that can be accessed by a network management protocol; for, the database maintained by a gateway running SNMP.

management station

The workstation of a network manager running SNMP.

mask

A means of subdividing networks using address modification. A mask is a dotted quad specifying the bits of the destination that are significant. Except when used in a route filter, GATED supports only contiguous masks.

mask length

The number of significant bits in the mask.

master file directory (MFD)

The root of an OpenVMS file system on a particular physical device.

master server

The name server that is the authority for a specific domain space. *See also* **BIND server**.

maximum transmission unit (MTU)

The largest possible unit of data that can be sent on a given physical medium. *See also* **fragmentation**.

MBUFs

See **memory buffers**.

memory buffers (MBUFs)

The portions of memory that act as queues for data arriving at a port before the process is ready to claim that data.

message

A message block or a series of message blocks that constitute a logical grouping of information; each is delimited by communications control characters.

metric

One of the units used to help a system determine the best route. Metrics may be based on hop count, routing delay, or an arbitrary value set by the administrator depending on the type of routing protocol. Routing metrics may influence the value of assigned internal preferences. *See also* **preference**.

MFD

See **master file directory**.

MIB

See **Management Information Base**.

MIB-II

See **Management Information Base II**.

MIME

Multipurpose Internet Mail Extensions; a specification for the transfer of nontext files with regular Internet e-mail.

mode

A protection placed on a file.

modem (modulator/demodulator)

A device that translates digital signals (electrical impulses) generated by a computer into analog signals (tones) that can be transmitted over telephone lines, and vice versa.

mount

An NFS process that makes a remote directory available to local users.

mount point

A directory on an NFS client that is associated with a remote file system. The directory must exist before NFS can use it as a mount point.

MTU

See **maximum transmission unit**.

multiaccess networks

Physical networks that support the attachment of multiple (more than two) routers. Each pair of routers on such a network is assumed to be able to communicate directly.

multicast

A transmission of network traffic intended for multiple hosts (but not all connected hosts) within a network or internet.

multicast address

An address that designates a subset of nodes that are all listening for packets destined to this address.

multicast addressing

An addressing mode in which a data packet is targeted to a group of nodes that are of the same type, for example, all level 1 routers or all level 2 routers.

multihomed host

A host that has two or more hardware connections to a network; requires multiple IP addresses.

multiplexing

Using a single connection to carry several data streams and the mechanism for assigning these streams to that connection.

multipoint circuit

A circuit that connects multiple systems.

multiprocessing system

A network consisting of multiple processors.

MX record

See **Mail exchange record**.

NAK

See **negative acknowledgment**.

name resolution

The process of mapping a host name to its corresponding address. *See also* **Domain Name System**.

named

The BIND Name Server daemon.

namespace

A commonly distributed set of names in which all names are unique.

negative acknowledgment (NAK)

The response to receipt of a corrupted packet of information. *See also* **acknowledgment**.

neighbor

Another router with which implicit or explicit communication is established by a routing protocol. Neighbors are usually on a shared network, but not always. This term is mostly used in OSPF and EGP. Usually synonymous with **peer**.

neighboring routers

Two routers that have interfaces to a common network. On multiaccess networks, routers are dynamically discovered by OSPF's HELLO protocol.

network

A group of computer systems that can communicate with each other; can be composed of computers in a single building (local area networks, or LANs), or computers thousands of miles apart (wide area networks or WANs). The Internet is a worldwide collection of computer networks that can intercommunicate.

Any packet-switched network. A network may be specified by its IP address or network name. The host bits in a network specification must be zero. Default may be used to specify the default network (0.0.0.0).

The IP address of a network. Usually specified as a dotted quad, with one to four values in the range of 0 through 255, separated by dots (.); for example, 132.236.199, 132.236, or 10. It can also be specified as a hexadecimal string preceded by 0x with an even number of digits between 2 and 8; for example, 0x??????, 0x???? or 0x0a. Also allowed is the symbolic value default that has the value 0.0.0.0, the default network. If options `noresolv` statement is not specified, this can also be a symbolic network name. For example, `nr-tech-prod`, `cornellu-net`, and `arpanet`. The numeric forms is preferred over the symbolic form.

network address

A unique identifier of a specific system on a network, usually represented as a number or series of numbers. *See also* **IP address**.

network architecture

The specification of a network's functions and its parts, together with the ways in which the network is organized; specifies the layers of different functions in the network, ranging from data transmission at the lowest levels to user applications at the highest levels.

network byte order

The order in which bytes of information are sent or received by network applications as opposed to how the bytes are stored in memory by different operating systems and hardware architectures. The standard network byte order is big endian.

network class

A definition of the type of network addressing scheme being used; high-order bits in the network number designate the network class of the IP address.

network database

The Compaq TCP/IP Services for OpenVMS database that allows users to refer to networks by name rather than network number; contains network names, IP addresses for the networks, and any alias names for the networks.

network delay

The time it takes to get a unit of data from the source of a transmission to the destination; usually refers to delay from the network and not by system-dependent application processing delays at source and destination nodes.

A HELLO metric. Valid values are from 0 to 30000, inclusive. The value of 30000 is the maximum metric and means unreachable. *See also* **metric** and **HELLO**.

network diameter

The distance (number of hops) between the two nodes in the network with the greatest reachability distance. The reachability distance is the path with the fewest number of hops between two nodes.

Network File System (NFS)

A protocol developed by Sun Microsystems that allows a computer system to access files over a network as if they were on its local disks.

Network Information Service (NIS)

A set of services in the Network File System that propagate information out from masters to recipients; used for the maintenance of system files on complex networks.

Network Interface

A device driver that communicates with the IP layer of the TCP/IP protocol suite and the network interface card.

Network Interface layer

The layer in the TCP/IP architecture model that 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 subnet in the IP address; each bit that is turned on (binary one) in the mask is interpreted as part of the network and subnet address. Synonymous with **subnet mask**.

A means of subdividing networks using address modification. A mask is a dotted quad specifying the bits of the destination that are significant. Except when used in a route filter, GATED supports only contiguous masks.

network meltdown

The state of complete network overload; the network equivalent of thrashing. *See also* **broadcast storm**.

network number

The part of an IP address that designates the network to which the destination host belongs.

network performance

The description of how a network performs, as measured against the expectations or requirements of users, customers, designers, or implementors, or as claimed by sales and marketing personnel. The criteria for network performance include parameters such as throughput, response time, and resource utilization.

network status notification

Information about the state of logical and physical links over which two tasks communicate. A nontransparent task can use this information to take appropriate action under conditions such as third-party disconnections and a partner's exiting before I/O completion.

network task

A nontransparent task that can process multiple inbound connection requests; that is, it has a declared network name or object number.

Network Time Protocol (NTP)

The protocol that ensures accurate local timekeeping with reference to radio and atomic clocks located on the Internet; capable of synchronizing distributed clocks within milliseconds over long time periods.

NFS

See **Network File System**.

NFS client

The software that requests remote file services from an NFS server. Client system users access files that physically reside on an NFS server system.

NFS server

The software that provides remote file services to NFS clients.

NFS server (OpenVMS server)

A computer system that offers services to NFS clients within an Internet environment; can be a single host, a whole OpenVMS Cluster system, or members of an OpenVMS Cluster system.

NIS

See **Network Information Service**.

nobody

A UNIX convention used when file ownership is not known; maps to an account with a UID and GID of -2.

node

(1) A system on a network; also referred to as a host. (2) One member in an OpenVMS Cluster system.

node address

The required unique numeric identification of a specific node in the network.

node name

The alphanumeric identification associated with the node address for one-to-one mapping.

nonadjacent nodes

Nodes without direct lines between them; can communicate only if intermediate systems forward the data along the path between the source and the destination.

nonauthoritative answer

A name server's answer is nonauthoritative when the server answer comes from its own cache.

nontransparent task

A form of device-dependent I/O that uses system services for network-specific functions; can initiate and complete a logical link connection, exchange messages between two tasks, and terminate the communication process. Application that has direct access to network-specific information and operations, such as optional user data on connects and disconnects and interrupt messages, to monitor the communications process; can receive and process multiple inbound connection requests.

normalization

The estimation of the change in a counter value over a specified time period.

nslookup

The Compaq TCP/IP Services for OpenVMS utility that allows you to interactively query domain name servers (BIND servers) and helps you set up and manage the BIND server software.

NTP

See **Network Time Protocol**.

NTP packet

A message sent over the network that conforms to the Network Time Protocol format. This format includes space for recording the current time. *See also* **poll**.

null modem

A simple form of modem connection in which only the data interchange circuits, not the modem control circuits, are used.

occluded mounting

A TCP/IP Services/NFS method of mounting an NFS file system onto a client mount point that is higher or lower in the directory structure than an active mount.

octet

A single, 8-bit unit of data. A networking term used instead of the term byte because some systems have bytes that are not 8 bits long.

ODS-2 disk structure

An OpenVMS On-Disk Structure. This is the default disk structure of the OpenVMS operating system.

ODS-5 disk structure (Alpha only)

An OpenVMS On-Disk Structure that is an extension to the existing ODS-2 disk structure. It adds the ability to use extended file names that can be more easily mapped between Windows and OpenVMS. ODS-5 expands the available character set and filename length to be consistent with Windows 95 and Windows NT. ODS-5 also supports deeper directories.

On-Disk Structure (ODS)

A logical structure given to information stored on a disk or CD-ROM. ODS is a hierarchical organization of files, their data, and the directories needed to gain access to them. The OpenVMS file system implements the On-Disk Structure and provides access control to the files located on the disk.

OPCOM

See **operator communication manager**.

OPCOM messages

Messages broadcast by the operator communication manager (OPCOM). These messages are displayed on operator terminals and written to the operator log file. The messages might be general messages that you send, user requests, operator replies, or system events.

OPCOM process

The system process that manages operator communication manager (OPCOM) operations.

open network

A network made up of nonproprietary, interoperable systems.

open network computing (ONC) remote procedure call (RPC)

An easy and popular paradigm for implementing the client/server model of distributed computing. In general, the local system (client) sends a request to a remote system (server) to execute a designated procedure, using supplied arguments, and the remote system returns the result to the local system.

operator communication manager

A system administration tool for communicating with users and operators on the system.

OSPF (Open Shortest Path First)

One of a class of interior gateway protocols, described in more detail in the OSPF section of `gated.proto(4)`.

open system

A nonproprietary, interoperable system with communications software.

Open System Interconnection (OSI)

A suite of protocols, designed by ISO committees, to be the international standard of computer network architecture.

OpenVMS Cluster

A configuration of OpenVMS processors in which the network sees the cluster as one system with one name, the cluster alias.

OpenVMS Cluster alias

An alias that allows remote hosts to address the cluster members as one host, as well as any cluster member individually.

OpenVMS file system

The OpenVMS files and directories on a mounted OpenVMS volume. These files and directories reside on a Files-11 On-Disk Structure (ODS-2 or ODS-5) disk.

origination

The beginning point of communications on a circuit.

overmounting

The process of NFS mounting another directory over an existing mount point. The original file system is dismounted from the mount point, and the new file system is mounted.

packet

A unit of data sent across a network.

Packet Internet Groper (PING)

A program used to test reachability of a destination by sending an ICMP echo request and waiting for a reply. *See also* **loopback**.

packet looping

A condition in which a packet revisits a node. *See also* **aged packet**.

packet size

The amount of data in a packet.

packet switching

A communication paradigm in which packets are individually routed between hosts, with no previously established communication path.

path

The physical lines between source nodes and destination nodes; can comprise a sequence of connected nodes. The path that the data takes through the network is transparent to users.

path cost

The sum of the circuit costs along a path between two nodes.

An OSPF (Open Shortest Path First) protocol metric. *See* **metric** and **OSPF**.

path length

The total distance (the number of circuits) between a source node and a destination node, measured in hops. Each line between systems, including routing nodes and end nodes, equals one hop. *See also* **network diameter**.

path name

A unique designation that identifies a directory or subdirectory. UNIX path names are composed of a series of fields separated by slashes (/); each field designates a file name that is uniquely contained in the previous field (directory).

path MTU

The smallest MTU of any data link that packets traverse between two hosts. The path MTU depends upon the route being used at the time. Therefore, the sending path MTU may differ from the receiving path MTU.

path MTU discovery

A mechanism to determine the path MTU at any one time.

path splitting

The ability to split the transmission load destined for a single node over several paths of equal path cost. Any destination node receiving data that has been split over several paths must support out-of-order packet caching.

PC-NFS Daemon

The server software that handles authentication and printing requests from personal computer implementations of NFS.

peer

Another router with which implicit or explicit communication is established by a routing protocol. Peers are often on a shared network. This term is used mostly by Border Gateway Protocol (BGP). Usually synonymous with **neighbor**.

physical address

A unique address of each physical connection of a node to the physical medium.

physical connection

The Physical layer communications path between two systems.

physical connectivity

The Physical layer connectivity that is a result of nodes being attached to each other via active lines and nodes.

PING

See **Packet Internet Groper**.

point-to-point circuit

A circuit that connects only two nodes. A point-to-point configuration requires a separate physical connection between each pair of nodes. Point-to-point systems communicate directly with other systems. *Contrast with* **multipoint circuit**.

point-to-point line

A line that connects two systems by using a single circuit.

Point-to-Point Protocol (PPP)

A method for transmitting datagrams over serial point-to-point lines where a line is established between a remote host (usually over a telephone line) and another host acting as a gateway to a remote host.

poll

The sending of an NTP packet from a host to an NTP time server to request the current time. The server responds by recording the current time in the packet, then sending it back to the originating host. *See also* **NTP packet**.

polling

Connecting to another system to check for things such as mail or news.

POP

See **Post Office Protocol**.

port

The endpoint of a communication link between two processes.

A UDP or TCP port number. Valid values are from 1 through 65535.

port number

A 16-bit number used to identify applications using TCP or UDP. The number is stored in the Transport layer protocol headers to identify the application.

Portmapper Service

A service that client programs can use to determine the port number that another service uses. Clients use the Portmapper Service for NFC, PC-NFS, and RPC applications.

post

To send a message to a mailing list or newsgroup. Distinguished in context from **mail**.

Post Office Protocol (POP)

The TCP/IP-based protocol for client stations to read mail from a server.

PPP

See **Point-to-Point Protocol**.

PPP client

A host requiring a temporary PPP connection to a dialup provider or a terminal server.

PPP dialup provider

A host that answers modem calls from PPP clients, assigns IP addresses and establishes PPP connections initiated by PPP clients.

preference

A preference is a value from 0 to 255 used to select a route from many routes to the same destination. The route with the best (numerically lowest) preference is selected as the active route. The active route is the one installed in the kernel forwarding table and exported to other protocols. Preference zero is usually reserved for routes to directly attached interfaces. A default preference is assigned to each source from which GATED receives routes.

prefix

A contiguous mask covering the most significant bits of an address. The prefix length specifies how many bits are covered.

primary server

A BIND name server that maintains the database for a zone; secondary servers copy their information from primary servers. Also called *primary master* or *master server*. *See also* **BIND server**, **cache server**, **forwarder server**, and **secondary server**.

printcap database

The Compaq TCP/IP Services for OpenVMS database that maps local queues to printers on remote hosts; specifies local queues for LPD printing from remote hosts. Equivalent to the UNIX `/etc/printcap` file.

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.

process

The context within a system in which a specific computing session occurs; provides the context in which an application executes.

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, such as the order in which the bits from a byte are set across a wire, or high-level exchanges between applications programs such as the way in which two programs transfer a file across the Internet.

protocol data unit (PDU)

The unit of data sent across a network. Also called a **packet**.

protocol machine

The set of data structures and routines that implements a specific protocol and controls the progress of a communication between peer entities.

protocol overhead

The part of communications data or processing not directly consumed by the users but necessary to successfully bring about the transfer of user information.

protocol port

An abstraction that transport protocols use to distinguish among multiple destinations within a given host computer. Internet protocols identify ports using small positive integers. Usually the operating system allows an application program to specify which port it wants to use. Some ports are reserved for standard services such as electronic mail.

protocol transparency

The quality in a communications device or system that allows various higher-level protocols to coexist on the same wire. The protocols are transparent to the device or system.

The degree to which users of underlying protocols are aware of the specifics of those protocols.

protocol sequence

An ordered list of protocol identifiers.

protocol stack

The set of functions, one at each layer of the protocol stack, that work together to form a set of network services; each layer of the protocol stack uses the services of the module beneath it.

proxy

The mechanism whereby one system acts on behalf of another system in responding to protocol requests. uses a proxy mechanism to provide an OpenVMS identity (account) for each UNIX client by adding the name and identification codes of the client to a proxy database.

proxy ARP

The technique in which one machine, usually a router, answers Address Resolution Protocol (ARP) requests intended for another machine. By "faking" its identity, the router accepts responsibility for routing packets to the "real" destination. Proxy ARP allows a site to use a single IP address with two physical networks. Normally, creating a subnet is a better solution.

proxy database

The database that provides OpenVMS identities for remote NFS clients and UNIX-style identities for local NFS client users; provides proxy accounts for remote processes.

pseudodevice

A software device used to implement special-purpose transports and not directly associated with hardware.

pseudointerface

A method of extending subnet routing using a network interface. Each network interface has one name and at most nine pseudointerface names. Each network interface and pseudointerface has its own IP address, network mask, and broadcast mask.

public domain

Intellectual property available to users that does not require payment of a fee.

quality of service (QoS)

The OSI equivalent of TOS.

RARP

See **Reverse Address Resolution Protocol**.

RCD

See **RMT/RCD**.

RCP

See **remote copy program**.

reachable node

The node to which the local node has a usable communications path.

read access

The access right that grants the ability to view data.

reassembly

The process of piecing together datagram fragments to reproduce the original datagram based on the fragmentation data in the IP header of the datagram.

reassembly time

A routing parameter that can be set to specify the length of time allowed for the reassembly of a message received in fragments. If the reassembly time expires before all fragments are received, the fragments are discarded.

Record Management Services (RMS)

The OpenVMS data management subsystem that defines the rules that govern the internal organization of and the methods of accessing file data.

reliability

The ability of a protocol to recover data that is damaged, lost, duplicated, or delivered out of order.

relative path name

A path name that does not start at the root; default directory is merged with the relative path name to form the absolute path name.

remote boot (BOOTP)

The software that supports the downloading of system images and other types of files to requesting clients.

remote copy program (RCP)

The program based on the Berkeley UNIX (*see* **BSD**) `rcmd` protocol that permits files to be copied from one computer to another by an extension to the syntax of the UNIX `cp` (copy) command. (RCP) does not provide the word-length adaptability and flexibility that the FTP protocol does.

remote line printing (LPR/LPD)

The remote printing services for UNIX and OpenVMS client hosts.

remote node

A node in the network other than the local node.

remote file system

A file system that resides on a network host other than the local node.

remote procedure call (RPC)

A programming interface for implementing the client/server model of distributed computing. In general, a request is sent to a remote system to execute a designated procedure, using arguments supplied, and the result returned to the caller. *See also* **ONC RPC**.

remote shell

A program that sends a command, shell, script, or command procedure to a remote host for execution.

remote task

A task either executing or originating at a remote host.

repeater

A bidirectional device that amplifies or synchronizes signals into standard voltages, currents, and timing; propagates electrical signals from one Ethernet to another without making routing decisions or providing packet filtering; Physical layer intermediate system. *See also* **bridge** and **router**.

Request for Comments (RFC)

A series of documents, begun in 1969, that describes the Internet suite of protocols and related experiments. Very few RFCs describe Internet standards, but all Internet standards are written as RFCs.

resolver

A mechanism or process to correlate a network host name into an appropriate network address in support of network applications; a network name resolver. *See* **BIND resolver**.

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.

resynchronization

A process that enables the recovery of user information lost or corrupted during transfer across an association. Sets the association back to the state it was in at a specified point in the transfer.

retransmission

A method of error recovery in which stations receiving messages acknowledge the receipt of correct messages and, on receipt of incorrect messages, either do not acknowledge or acknowledge in the negative. The lack of acknowledgment or receipt of a negative acknowledgment indicates to the sending station that it should transmit the failed message again.

Reverse Address Resolution Protocol (RARP)

The TCP/IP protocol that provides the reverse function of ARP. This protocol maps a physical (hardware) address to an IP address. Often used by diskless nodes when they first initialize to find their Internet address.

reverse domain

An Internet domain that BIND servers use to map IP addresses to domain names.

RFC

See **Request for Comments**.

RFC 822

The TCP/IP standard format for electronic mail message headers; often referred to as "822 messages". The name comes from RFC 822 that contains the specification; previously known as 733 format.

RIB (routing information base)

routing database

RIP

See **Routing Information Protocol**.

rlogin

Remote login: The Berkeley 4.3 BSD service that allows users of one machine to connect to other systems across the Internet and interact as if their terminals are connected the machines directly.

RMS

See **Record Management Services**.

RMT/RCD

Remote command that allows remote users to access magnetic tapes and CD drives.

root

The top level directory in a UNIX-style file system; 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.

root server

An Internet name server that knows about all of the top-level domains on the Internet network; the master servers for the Internet root zone.

round-trip delay

The total time during communications that implement a protocol with positive acknowledgments, for a message to be transmitted, arrive at its destination, and its corresponding acknowledgment to be sent and subsequently received by the sender of the original message.

The time it takes for a host to send an NTP packet to another host and get an NTP packet back from that host in reply.

round-trip time (RTT)

A variable computed during TCP sessions that indicates the total time required to send a TCP segment to a remote host and receive a reply.

route

The path over the network that information takes to get from one source to its destination.

route through

Data packets not destined for the local node.

routes database

The Compaq TCP/IP Services for OpenVMS database that specifies Internet gateways.

ROUTED

See **Routing Daemon**.

Routing Daemon (ROUTED)

A program that runs under 4.2BSD/4.3BSD UNIX systems (and derived operating systems) to propagate routes among machines on a local area network using the Routing Information Protocol; pronounced "route-dee."

One of a class of interior gateway protocols, described in more detail in the RIP section of gated.proto(4).

router

A node that can send and forward data to and receive data from other nodes.

router advertisement

A Router Discovery Protocol message sent out by Router Discovery Servers to announce their existence to hosts. The router advertisement contains a list of all router addresses on a given interface and their preferences for use as a default router.

Router Discovery Protocol

An IETF standard protocol used to inform hosts of the existence of routers. It is used in place of or in addition to statically configured default routes in hosts. The protocol has a server portion that runs on routers, and a client portion that runs on hosts.

router id

A 32-bit number assigned to each router running the OSPF protocol. This number uniquely identifies the router within the autonomous system.

router_id

An IP address used as unique identifier assigned to represent a specific router. This is usually the address of an attached interface.

router solicitation

A Router Discovery Protocol message sent out by a host to request router advertisement responses from a router.

routing

A Network layer function, implemented in intermediate systems, that determines the path along which data travels to its destination and the movement of that data. *See also* **decision**.

routing database

The database that contains routing information, including destination host names, IP addresses for the hosts, gateway host names, and IP addresses for the gateways. There are two route databases: the static route database that is maintained on disk, and the volatile database in memory.

The repository of all of **GATED**'s retained routing information, used to make decisions and as a source for routing information that is propagated.

routing domain

A set of hosts and routers within a single administrative domain that operates according to the same routing procedures.

Routing Information Protocol (RIP)

The protocol that enables gateways to broadcast their current routing database to hosts and networks that are connected directly to them. software implements the RIP through its dynamic routing Zserver.

One of a class of interior gateway protocols, described in more detail in the RIP section of `gated.proto`(4).

routing protocol

A protocol sent between routers by which routers exchange information on how to route to various parts of the network. The TCP/IP family of protocols has many of this type of protocol, such as RIP, EGP, BGP, OSPF, and dual IS-IS.

routing socket

A data structure used by processes to communicate routing information to the kernel. A process can add and delete routes, dump the routing table, and read messages from the kernel. The only type of socket supported in the `AF_ROUTE` domain is a raw socket.

routing table

The repository of all of `gated`'s retained routing information, used to make decisions and as a source for routing information that is propagated.

RPC

See **remote procedure call** and **ONC RPC**.

rshell

Remote shell; a remote utility that enables the user to open a shell session on a remote host.

RTL

See **run-time library**.

RTT

See **round-trip time**.

run-time library (RTL)

A collection of OpenVMS procedures available to native mode images at run time; provide support routines for high-level language compilers.

SCALE

A TCP window scaling option; allows window information to be interpreted as being scaled by 1 to 16 powers of 2, thus increasing the size of the effective window.

secondary server

A master BIND server that receives authoritative database information from a primary server. Also known as *slave server*. *See also* **BIND server**, **cache server**, **forwarder server**, and **primary server**.

segment

A unit of data exchanged by the TCP modules.

segment length

The amount of sequence number space occupied by a segment, including controls that 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).

serial device

A device that uses serial transmission; that is, transmits data one bit at a time on a single channel as opposed to parallel transmission, which transmits one or more bits at a time on one or more channels. Typically, terminals and printers are serial devices.

Serial Line Internet Protocol (SLIP)

A protocol designed to allow a host to connect to another host over serial lines, such as telephone circuits or RS-232 cables.

server

A process that offers a service to another process over the network and accepts requests from other processes, known as **clients**.

service

(1) A task that an application can carry out. (2) The interface provided by a service element or layer for accessing one or more function.

service interface

The boundary at which a layer provides a service to the adjacent higher layer in the network architecture; may vary between implementations.

service parameter

The means by which a service user and a service provider exchange information.

service provider

In network architecture, the service element or layer that provides a set of services to the layer immediately above.

service specification

An international standard that describes the functions and service parameters of every service of a service provider.

service user

An application program, service element, or Network layer that uses the services of a service provider.

services database

The Compaq TCP/IP Services for OpenVMS database created by default that contains one entry for each service configured.

Simple Mail Transfer Protocol (SMTP)

An Internet standard protocol for transferring electronic mail messages from one machine to another; 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; allows remote monitoring and management of network devices (particularly routers and servers) from across an Internet.

simplex

An interface may be marked as simplex either by the kernel or by the interface configuration. A simplex interface is an interface on a broadcast medium that is not capable of receiving packets it broadcasts.

The **GATED** daemon takes advantage of interfaces that are capable of receiving their own broadcast packets to monitor whether an interface appears to be functioning properly.

skew

A measure, in Hertz, of the difference between the actual frequency of a clock and what its frequency should be to keep perfect time. *See also* **drift**.

slave server

A name server that has no access to the Internet and relies on forwarder servers to resolve queries that it cannot resolve locally. As slave servers receive information from forwarder servers, they store that information in their cache. *See also* **cache server**, **forwarder server**, **primary server**, and **secondary server**.

slew

To adjust gradually the time of a clock until it tells the correct time. *Compare with* **step**.

SLIP

See Serial Line Internet Protocol

SMI

See **Structure of Management Information**.

SMTP

See **Simple Mail Transfer Protocol**.

SNMP

See **Simple Network Management Protocol**.

socket

The endpoint of communication to which an IP address and port may be bound. When writing an application, it is a data structure that is part of the Internet pseudodevice created every time an OpenVMS process assigns a communication channel. The other part of the Internet pseudodevice is the device socket.

socket API

An application programming interface for implementing TCP/IP protocols. Sometimes called Berkeley Sockets indicating where the API was developed.

socket pair

The client IP address and port number, and the server IP address and port number that uniquely identify a TCP connection.

source

The IP header field that contains the IP address of the datagram's point of origin.

source port

A 2-octet value in the TCP or UDP header field that identifies the upper-level application or protocol associated with the data in the segment.

spanning tree

A logical arrangement created by bridges in an extended LAN in which all LANs are connected and there are no loops.

split horizon

When a router (or group of routers work together) accepts routing information from multiple external networks, but does not pass on information learned from one external network to others. This is an attempt to prevent false routes to a network from being propagated because of gossip or counting to infinity.

splitting

The process of mapping one transport connection to several network connections.

stateless

A characteristic of a server designed to simplify crash recovery after a server crashes and reboots. The server does not keep track of the status of ongoing client interactions. Servers that do not keep track of client status are called stateless servers.

static routing

A routing method by which a system manager manually adds routes to the kernel's routing table. This method is generally used on small networks. On Open VMS systems, you use the SET ROUTE command to add static routes and on UNIX systems, you use the `route` command.

step

To change the time of a clock to the correct time with no intermediate adjustments. *Compare with* **slew**.

stratum

The distance a host running the NTP time daemon is from an external source of Coordinated Universal Time (UTC). A stratum 1 server has direct access to an external source of UTC, such as a radio clock synchronized to a standard time signal broadcast. In general, a stratum *n* server is *n* - 1 network hops away from a stratum 1 server. For example, a stratum 4 server is 3 hops away from a stratum 1 server. Also, a stratum *n* server is at a higher stratum than a stratum *n* - 1 server. For example, a stratum 3 server is at a higher stratum than a stratum 2 server, and at a lower stratum than a stratum 4 server. *See also* **time daemon**.

stream-oriented

The type of transport service that allows its client to send data in a continuous stream; guarantees that all data will be delivered to the other end in the same order as sent and without duplicates. Also known as a reliable transport service.

Structure of Management Information (SMI)

The rules used to define the objects that can be accessed by means of a network management protocol. *See also* **Management Information Base**.

subnet

An organization of hosts within a network into logical groups. A network can be comprised of several subnets. The portion of a network, which might be a physically independent network, that shares a network address with other portions of the network and is distinguished by a subnet number. A subnet is to a network what a network is to an internet.

subnet address

A part of the Internet addressing scheme. If a site uses a single IP address for multiple physical networks, there is one subnet address for each physical network. Each such address is composed of the network part of the full address and part of the local part (host).

subnet field

A bit field in an IP address that denotes the subnet number. The bits making up this field are not necessarily contiguous in the address.

subnet mask

A method of representing the portion of the IP network address that is devoted to subnet address. Each bit that is turned on (binary one) in the mask is interpreted as part of the network and subnet address. Synonymous with network mask. *See* **address mask**.

superuser

A UNIX user who has been granted special privileges; has an effective UID of 0.

symbiont

A process that transfers record-oriented data to and from a mass storage device; for example, from disks to printers.

Synonym for daemon.

symbolic link

In the UNIX file system, a symbolic link is a file that contains a pointer to another file or directory. The link (also called a soft link) may be created across a different UNIX file system. Any changes to the file can be seen when you access the file through the file name or through the symbolic link. If you delete the file, the symbolic link will point to a nonexistent file.

synchronous transmission

Data transmission in which characters are transmitted at a fixed rate. The transmitter and receiver are synchronized, gaining greater efficiency than in asynchronous transmission. Synchronous transmissions send a predetermined group of "sync" characters ahead of a long stream of data. The sync characters enable the communicating devices to synchronize with each other in accordance with a time clock at each end. *Contrast with* **asynchronous transmission**.

syntax

The rules for formatting or interpreting data.

TAC

See **terminal access controller**.

target system

The intended destination of messages.

TCP

See **Transmission Control Protocol**.

TCP/IP

An Internet suite of protocols. *See also* **Transmission Control Protocol** and **Internet Protocol**.

TELNET

An Internet protocol for remote terminal connection. TELNET allows a user at one site to interact with remote timesharing systems at another site as if the user's terminal were directly connected to the remote host.

terminal access controller (TAC)

A program and hardware that connects terminals to the Internet, usually using dialup modem connections.

terminal emulator

A program that allows a computer to emulate a terminal; a workstation thus appears as a terminal to the host.

terminal server

A device that handles terminal operations for host nodes on a LAN; can be used to connect terminal users to nodes on the same LAN and to users on nodes located off the LAN. Offloads the terminal connection and I/O responsibilities from host nodes, and reduces the number of direct terminal connections to each host, thus saving substantial power, packaging, and cabling expense.

terminating packet

A packet whose destination is the local node.

TFTP

See **Trivial File Transfer Protocol**.

thread

(1) A request from an NFS client to the NFS server. (2) A single unit of execution within a program.

throughput

A measure of how much data is sent, or can be sent, between two points in a specified unit of time; often used in either of two contexts:

- Rated throughput, which refers to the bandwidth or capacity of a component.
- Real throughput, which refers to actual measured throughput.

time

A time value, usually a time interval that can be specified in any one of the following forms:

```
number
A non-negative decimal number of seconds. For example, 27, 60, or
3600.

number:number
A non-negative decimal number of minutes followed by a seconds value
in the range of zero to 59, inclusive. For example, 0:27, 1:00, or
60:00.

number:number:number
A non-negative decimal number of hours followed by a minutes value in
the range of zero to 59, inclusive, followed by a seconds value in
the range of zero to 59, inclusive. For example, 0:00:27, 0:01:00,
or 1:00:00.
```

time to live (TTL)

A field in the IP header that indicates how long this packet should be allowed to be forwarded to other routers before being discarded.

The Time To Live (TTL) of an IP packet. Valid values are from 1 to 255 inclusive.

time daemon

The program running on a host that synchronizes the host's hardware clock to Coordinated Universal Time in accordance with the protocols known as the Network Time Protocol.

timeo

A timeout option for the NFS `mount` command.

TN3270

TELNET options that allows TELNET users to connect to hosts that support 3270 model terminals.

Token Ring

A type of LAN that has stations wired in a ring, where each station constantly passes a special message (a "token") on to the next; technically referred to as IEEE 802.5.

topology

The architecture of a network. A network topology shows the computers and the links between them within a network.

TOS (type of service)

An IP header field that specifies the importance of a datagram and how to make tradeoffs between delay, throughput, reliability, and cost when the datagram travels across a network. The parameters are mapped into actual service parameters for the particular networks the datagram crosses.

traffic

The measurement of data flow, volume, and velocity over a communications link.

transceiver

Transmitter-receiver; a physical device required in baseband networks that takes the digital signal from a computer or terminal and imposes it on the baseband medium; connects a host interface to a LAN, such as Ethernet.

transient information

Network management information carried in an operation; is meaningful only while the operation is being performed.

transit network

A network that passes traffic between networks in addition to carrying traffic for its own hosts; must have multiple connections to the internet.

Transmission Control Protocol (TCP)

A Transport layer protocol that provides the reliable, full-duplex, stream service on which many application protocols depend. TCP allows a process on one host 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.

Transmission Control Protocol/Internet Protocol (TCP/IP)

The acronym for the suite of application and transport protocols that run over IP, such as FTP, TELNET, and UCP, as well as TCP and IP.

Transport layer

The layer in the TCP/IP architecture model where network traffic is passed between an application on one host and an application on another host.

Trivial File Transfer Protocol (TFTP)

The Internet protocol for file transfer with minimal capability and minimal overhead. The simple design of the facility is intended for use in application environments that do not require complex interactions among clients and servers. TFTP is a simple service running on top of UDP, using timeout and retransmission to ensure that data arrives. The sending side transmits a 512-byte, fixed-size file, and awaits an acknowledgment for each block before sending the next. The receiver acknowledges each block. *See also* **File Transfer Protocol**.

TTL

See **time to live**.

tunneling

The encapsulation of protocol A within protocol B such that A treats B as though B were a Network Interface layer. Used to get data between administrative domains that use a protocol not supported by the internet connecting those domains.

UAF

See **user authorization file**.

UCP

See **Management Control Program**.

Management Control Program

The Compaq TCP/IP Services for OpenVMS network management control software; includes a command-line interface.

UDP

See **User Datagram Protocol**.

UID

See **user identification**.

UNIX-style file system

An OpenVMS organization of files based on the UNIX operating system. Also known as a container file system.

UNIX-to-UNIX Copy Program (UUCP)

A program that allows one UNIX system to copy files to or from another UNIX system.

upline dumping

A TFTP server function allowing a TFTP client to transfer data or a program image to the TFTP server's public directories. The opposite function of downline loading.

user authorization file (UAF)

An OpenVMS file that contains account names and their associated attributes.

User Datagram Protocol (UDP)

An Internet transport protocol. A connectionless, unreliable Transport layer protocol for the exchange of requests and replies between networked hosts. UDP, like TCP, uses IP for message delivery from one host to another; however, unlike TCP, UDP provides for exchange of datagrams without acknowledgments or guaranteed delivery of data. Each UDP message contains the data sent by a user process, a destination port number, and a source port number.

user identification (UID)

A unique number that identifies a user of a UNIX system. The number along with an associated group identification number (GID) determines file access privileges. UID also tracks accounting statistics and other collected information.

UUCP

See **UNIX-to-UNIX Copy Program**.

virtual circuit

The network service that allows two processes to communicate as if they were directly connected, regardless of the structure of the underlying subnet.

WAN

See **wide area network**.

well-known port

A port number assigned for use by a specific network application for connections made with either UDP or TCP. Every implementation of TCP/IP that provides well-known services provides them with the well-known port numbers from 1 to 1023. The Internet Assigned Numbers Authority (IANA) manages the well-known port numbers.

wide area network (WAN)

A network, usually constructed with serial lines, which covers large geographic areas.

wildcarding

A method for generalizing parts of a OpenVMS file designation to encompass a set of files by substituting a symbol to represent one or more characters. OpenVMS wildcarding symbols are the percent sign (%) for a single character, and the asterisk (*) for a character string of any length, including zero.

window

A 2-octet field in a TCP header indicating the number of data octets (relative to the acknowledgment number in the header) that the sender is currently willing to accept.

write access

An access right that grants users the ability to change data.

zone

A subdivision of the Internet hierarchy that starts at a domain and extends down to leaf domains (individual host names) or to domains where other zones begin; usually represents an administrative boundary. *Contrast with* **domain**.

zone file

A master name server file that describes the domain names for which the server has authority.

Acronyms

The following table shows Compaq TCP/IP Services for OpenVMS acronyms and other acronyms related to open networking:

Acronym	Meaning
ACK	acknowledgment
ACL	access control list
ACP	ancillary control process
ANSI	American National Standards Institute
API	application programming interface
ARP	Address Resolution Protocol

Acronym	Meaning
ASCII	American Standard Code for Information Interchange
ATM	asynchronous transfer mode
BBS	Bulletin Board System
BGP	Border Gateway Protocol
BIND	Berkeley Internet Name Domain
BOOTP	Bootstrap Protocol
bps	bits per second
BSD	Berkeley Software Distribution
CSLIP	Compressed Serial Line Internet Protocol
DCE	Distributed Computing Environment
DCL	DIGITAL Command Language
DEK	data encryption key
DES	data encryption standard
DNS	Domain Name System
eSNMP	extensible Simple Network Management Protocol
EGP	External Gateway Protocol
FDDI	Fiber Distributed Data Interface
EOF	end of file
EOL	end of line
FQDN	fully qualified domain name
FTP	File Transfer Protocol
GID	group identification (UNIX)
IAB	Internet Architecture Board
ICMP	Internet Control Message Protocol
IGP	Internal Gateway Protocol
InterNIC	Internet Network Information Center
IP	Internet Protocol
ISDN	Integrated Services Digital Networks
IVP	installation verification procedure
Kbps	kilobits per second
LAN	local area network
LPD	line printer daemon
LPR	remote line printing
MBUF	memory buffer
MCP	Management Control Program
MFD	master file directory
MIB	Management Information Base
MIB-II	Management Information Base II

Acronym	Meaning
MTU	maximum transmission unit
MX	Mail exchange
NAK	negative acknowledgment
NFS	Network File System
NIS	Network Information Service
NOC	Network Operations Center
NTP	Network Time Protocol
PDU	protocol data unit
PING	Packet Internet Groper
POP	Post Office Protocol
PPP	Point-to-Point Protocol
PSDN	Packet Switching Data Network
PWIP	PATHWORKS Internet Protocol
RARP	Reverse Address Resolution Protocol
RCP	remote copy
REXEC	remote execute
RFC	Request for Comments
RLOGIN	remote login
RIP	Routing Information Protocol
RMS	Record Management Services
RPC	remote procedure call
RSH	remote shell
RTL	run-time library
RTT	round-trip time
SLIP	Serial Line Internet Protocol
SMI	structure of management information
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
TAC	terminal access controller
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TFTP	Trivial File Transfer Protocol
TP	Time Protocol
TTL	time to live
UAF	user authorization file

Acronym	Meaning
UDP	User Datagram Protocol
UID	user identification (UNIX)
UTC	Coordinated Universal Time
UUCP	UNIX-to-UNIX Copy Program
WAN	wide area network
WKS	well-known server
XDR	external data representation

Index