

TCP/IP Services for OpenVMS

Student Guide

Training



COMPAQ

TCP/IP Services for OpenVMS

Student Guide

Training



COMPAQ

Notice

© 1999 Compaq Computer Corporation

COMPAQ, the Compaq logo, and the Digital logo are registered in the U.S. Patent and Trademark Office. Other product names mentioned herein may be the trademarks of their respective companies.

Confidential computer software. Valid license from Compaq or authorized sublicensor required for possession, use, or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Compaq Computer Corporation shall not be liable for technical or editorial errors or omissions contained herein. The information in this document is subject to change without notice.

Aero, ALPHA, ALPHA AXP, AlphaServer, AlphaStation, Armada, BackPaq, COMPAQ, Compaq Insight Manager, CompaqCare logo, Counselor, DECterm, Deskpro, DIGITAL, DIGITAL logo, DIGITAL Alpha Systems, Digital Equipment Corporation, DIGITAL UNIX, DirectPlus, FASTART, Himalaya, InfoPaq, Integrity, LicensePaq, Ministation, NetFlex, NonStop, OpenVMS, PaqFax, Presario, ProLiant, ProLinea, ProSignia, QuickBack, QuickFind, Qvision, RDF, RemotePaq, RomPaq, ServerNet, SERVICenter, SmartQ, SmartStart, SmartStation, SolutionPaq, SpeedPaq, StorageWorks, Systempro/LT, Tandem, TechPaq, TruCluster, and Tru64 UNIX are registered in United States Patent and Trademark Office.

Atalla, C-Series, Expand, FOX, Guardian, iTP, Measure, Netelligent, and PointView are trademarks of Compaq Computer Corporation.

Microsoft, Windows, and Windows NT are registered trademarks of Microsoft Corporation.

Intel, Pentium, and Intel Inside are registered trademarks and Xeon is a trademark of Intel Corporation.

UNIX is a trademark in the US and other countries, licensed exclusively through X-Open Company Ltd.

TCP/IP Services for OpenVMS

Student Guide
December 1999

Course Overview

Course Description	Overview-1
Course Length	Overview-1
Intended Audience	Overview-1
Course Content and Objectives	Overview-1
Prerequisites	Overview-1
Course Conventions	Overview-1
Course Outline	Overview-2

Module 1 — Network Overview

Introduction	1-1
Objectives	1-1
Topics	1-1
Compaq TCP/IP Services for OpenVMS	1-2
Network Concepts	1-3
Network Communication Essentials	1-4
What Is a Protocol?	1-5
Physical Access to the Wire	1-6
Application Layer Protocol	1-8
Port Numbers	1-9
Media Access Control (MAC) Addresses	1-10
Routing and Network Addresses	1-12
Transport Layer (TCP)	1-14
Multiple Services	1-16
Transport Layer (UDP)	1-17
Protocol Stack	1-18
Sockets	1-19
The Internet	1-20
IP Addresses	1-21
Network ID and Host ID	1-22
Internet Addresses	1-23
IP Address Classes	1-24
Subnet Masks	1-25
Nondefault Subnet Masks	1-27
Class C Subnet Case	1-28
Broadcast Addresses — Class C Subnet Case	1-33

Module 2 — TCP/IP Commands

Introduction	2-1
Objectives.....	2-1
Topics.....	2-1
User Services.....	2-2
Remote Computing with Telnet	2-5
Proxy Accounts	2-6
Remote Computing with RCP.....	2-7
Remote Computing with RLOGIN.....	2-9
Remote Computing with RSH.....	2-11
Remote Computing with FINGER	2-12
File Transfer with FTP.....	2-13
File Transfer with TFTP	2-15
Resource Sharing.....	2-16
Network Services.....	2-16
Management Services.....	2-16
Management Examples.....	2-17

Module 3 — Installing and Configuring TCP/IP Services

Introduction	3-1
Objectives.....	3-1
Topics.....	3-1
Before Installation	3-2
Installation.....	3-3
Installation Sample	3-3
Displaying Information.....	3-4
Configuration.....	3-5
TCPIP\$CONFIG	3-6
Configuration Sample	3-7
Automatic Startup and Shutdown.....	3-29

Module 4 — Protocols

Introduction	4-1
Objectives.....	4-1
Topics.....	4-1
Data Link Protocol.....	4-2
Protocol Identification	4-3
Protocol Types.....	4-3
Ethernet Encapsulation	4-4
IP Header.....	4-5
TCP Header.....	4-6
CSMA/CD.....	4-7
Ethernet Addresses	4-8

Network Standards.....	4-10
Using the Protocol Stack.....	4-11
Ethernet Frame	4-12
IP Protocol Header.....	4-13
IP Protocol TOS Field.....	4-15
Internet Protocol Suite	4-16
TCP and IP Protocol	4-17
TCP Protocol.....	4-18
UDP	4-19
ARP and ICMP.....	4-21

Module 5 — Network File System (NFS)

Introduction	5-1
Objectives.....	5-1
Topics.....	5-1
NFS Concepts.....	5-2
Client/Server Strategy.....	5-4
NFS Implementation.....	5-5
NFS on OpenVMS.....	5-6
NFS Server File Systems	5-7
Selecting a File System.....	5-8
Container File System (CFS)	5-8
NFS Management Commands.....	5-9
Server Commands.....	5-9
Client Commands	5-9
Proxy Accounts	5-10
NFS Client Configuration	5-11
NFS Client Data Manipulation Examples.....	5-18
NFS Client File Ownership.....	5-19
NFS Client Daemon I/O.....	5-20
NFS Client Dismount and Mount Using Proxies	5-22
UNIX Server Handling OpenVMS File Attributes	5-23
Mount with Server Allowing root Access.....	5-24
NFS Server Configuration.....	5-26
NFS Server Data Manipulation Examples	5-29
Map Command	5-29
Exporting.....	5-30
Exporting and Mapping	5-31
File Attributes.....	5-32
NFS Server Container File Manipulation Examples	5-34
Mapping a UNIX Style Name to a Directory.....	5-36
Exporting.....	5-37

Module 6 — Serial Lines

Introduction	6-1
Objectives.....	6-1
Topics.....	6-1
Serial Connections.....	6-2
SLIP Encapsulation	6-3
PPP Encapsulation.....	6-4
PPP Configuration	6-5
Setting Up SLIP.....	6-6

Module 7 — Routing

Introduction	7-1
Objectives.....	7-1
Topics.....	7-1
Routing.....	7-2
Router or Gateway.....	7-3
Routing Table Decisions.....	7-4
Router Decision Process	7-5
Routing Table.....	7-6
Static Routing	7-8
TCPIP\$ROUTE.....	7-9
Displaying Static Routes.....	7-10
Dynamic Routing.....	7-11
Routing Daemon (ROUTED).....	7-12
Gateway Routing Daemon (GATED).....	7-13
GATED Supported Protocols.....	7-14
Enabling and Disabling Dynamic Routing	7-15
Configuring GATED	7-16
Sample GATED Configuration File	7-17
Configuring IP Forwarding.....	7-19
Configuring Routing Examples.....	7-20
Configuring Routing.....	7-20
Adding a Default Route	7-22
Forwarding	7-24
ROUTED	7-25
Dynamic Routing.....	7-26
Tracing ROUTED Traffic.....	7-30
Tracing ROUTED Communications	7-31
Fully Tracing ROUTED Traffic.....	7-32

Module 8 — DNS and BIND

Introduction	8-1
Objectives.....	8-1
Topics.....	8-1
Domain Name Service (DNS).....	8-2
DNS History	8-3
Early Internet.....	8-3
JEEVES and BIND.....	8-4
Hierarchical Name Format.....	8-5
Domains and Subdomains.....	8-6
DNS Zones	8-7
DNS Master and Slave Servers	8-8
Files and Resource Records	8-9
DNS Server Types	8-12
Master Server	8-12
Slave Server.....	8-12
Caching-Only Server	8-13
Forwarder Server.....	8-13
DNS Query Types.....	8-14
Recursive.....	8-14
Iterative	8-14
DNS Daemon Processes.....	8-15
Resolver Example.....	8-16
Resolver Configuration.....	8-19
NSLOOKUP	8-26
NSLOOKUP Internal Commands	8-27
NSLOOKUP Examples	8-28
BIND Master Server Setup	8-31
TCPIP\$CONFIG	8-31
Enable BIND.....	8-32
Configuration Files	8-34
TCPIP\$BIND.CONF	8-36
Forward Translation File.....	8-39
Reverse Translation Files.....	8-41
Log File.....	8-43
Server Access from Client	8-44
Looking at Resource Records	8-46
Correcting Database File Errors.....	8-48
Show Server Characteristics.....	8-49
Accessing Server Database Files.....	8-50
Adding a New Node	8-52

Cluster Load Balancing.....	8-54
Load Broker	8-55
Configuring the Load Broker and Metric Server	8-56
Load Broker and Metric Server Interaction	8-57

Module 9 — Bootstrap Protocol (BOOTP)

Introduction	9-1
Objectives.....	9-1
Topics.....	9-1
BOOTP (Bootstrap Protocol)	9-2
BOOTP Server	9-3
TFTP Download	9-4
Controller Activities	9-5
BOOTP Commands and Logicals	9-6
BOOTP Server	9-7
Adding Entries to the Server Database	9-7
Download File Location	9-7
Setting Up the BOOTP and TFTP Services.....	9-7
Enabling and Disabling BOOTP and TFTP	9-8
TFTP Characteristics	9-8
BOOTP Configuration and Data File	9-9
Enabling TFTP	9-13
Command Examples.....	9-14
TFTP File Configuration.....	9-16
BOOTP Data File Configuration.....	9-17
Trace During Ethernet Boot of Cuff (Tru64 UNIX V4.0e)	9-18
Trace During Ethernet Boot of Cuff with TFTP	9-20

Module 10 — Dynamic Host Configuration Protocol (DHCP)

Introduction	10-1
Objectives.....	10-1
Topics.....	10-1
Comparing DHCP and BOOTP	10-2
Dynamic Address Assignment	10-3
Relay Agent.....	10-4
DHCP Advantages.....	10-4
DHCP Configuration	10-5
DHCP Configuration Files.....	10-6
DHCPTAGS.....	10-9
NAMEPOOL, NETMASKS, and NETS	10-11
SERVER.PCY.....	10-13
TCPIP\$DHCP_SETUPCOMMANDS.COM	10-14
Process Details	10-17
Support Commands	10-17
BOOTP Interactions	10-18

GUI Examples	10–20
Parameters	10–20
Dynamic Addressing.....	10–21
Static Addressing.....	10–22
Hardware Types	10–22
Static and Permanent Address Assignment	10–23
Configuring DHCP Parameters.....	10–24
DHCP Cluster Failover Support.....	10–25

Module 11 — Network Time Protocol (NTP)

Introduction	11–1
Objectives.....	11–1
Topics.....	11–1
Time Synchronization.....	11–2
Network Time Synchronization	11–3
Time Sources.....	11–5
Configuration Modes	11–6
NTP Directory and Template	11–7
NTP Template File	11–9
NTP Shutdown and Startup.....	11–11
NTP Support Files	11–13
Drift File.....	11–13
Log File.....	11–13
NTPTRACE	11–14
NTPDC	11–15
NTPQ.....	11–17
TCPTRACE	11–19

Module 12 — Simple Network Management Protocol (SNMP)

Introduction	12–1
Objectives.....	12–1
Topics.....	12–1
Agents (Servers) and Management Stations (Clients).....	12–2
Agent (Server).....	12–3
NMS (Client).....	12–4
SNMP Communication.....	12–5
MIB.....	12–6
ASN.1	12–6
Object Names and Identifiers.....	12–7
SNMP Protocol Format	12–8
SNMP on OpenVMS	12–9
SNMP Components	12–10
SNMP Through TCPIP\$CONFIG.....	12–10
SNMP Startup and Shutdown	12–11

Log Files and Command Procedures.....	12-12
Changing Default Configuration	12-19
Showing Configuration.....	12-21
Support Commands	12-22
GetRequests.....	12-22
SetRequests	12-24
TCPTRACE of GetRequest	12-25
TCPTRACE of SetRequest.....	12-27
MIB-II	12-29

Module 13 — Remote Printing

Introduction	13-1
Objectives.....	13-1
Topics.....	13-1
Remote Printing Options.....	13-2
LPR/LPD	13-3
Configuring a Local Printer	13-4
Starting Up the Local Printer	13-10
Deleting the Printer.....	13-11
Reconfiguring the Local Printer	13-12
Configuring the Remote Printer	13-14
Remote Printing.....	13-18
TCPTRACE	13-20
Printer Access Control.....	13-23
TELNETSYM	13-24
PC-NFS	13-25

Module 14 — Troubleshooting

Introduction	14-1
Objectives.....	14-1
Topics.....	14-1
Configuration Troubleshooting	14-2
Network Troubleshooting	14-3
Common Sense Checks.....	14-3
Troubleshooting Hints	14-4
BOOTP Troubleshooting	14-4
TFTP Troubleshooting.....	14-4
FTP Troubleshooting	14-5
LPR/LPD Troubleshooting	14-6
NFS Troubleshooting.....	14-6
Lower-Level Protocol Testing.....	14-7
Network Interface Problems.....	14-8
Connectivity	14-9

Network Traffic Problems.....	14–10
Routing Problems	14–10
Dynamic Routing.....	14–11
DNS Problems.....	14–11
Load Broker Troubleshooting	14–11
Performance Tuning	14–12

Module 15 — Simple Mail Transfer Protocol (SMTP)

Introduction	15–1
Objectives.....	15–1
Topics.....	15–1
SMTP Concepts.....	15–2
SMTP Client Commands	15–2
SMTP Implementation.....	15–4
Post Office Protocol (POP)	15–7
Mail Addresses	15–8
Multipurpose Internet Mail Extensions (MIME)	15–9

Appendices

Appendix A — Binary Mask Templates
Appendix B — NSLOOKUP Internal Commands
Appendix C — Additional Routing
Appendix D — Wire World

Course Description

Course Length

Five days.

Intended Audience

This course is designed for computer professionals interested in understanding and administering the TCP/IP environment within the OpenVMS operating system.

Course Content and Objectives

This course is designed to provide you with the information and experience needed to function in a networked environment focused on the TCP/IP protocols. After presenting introductory information on protocols and standards, the course covers the components and services within TCP/IP. The central protocols of TCP/IP are examined with emphasis on their purpose. That is, what need do they fulfill in the big picture of node-to-node-communication? The course introduces user- and administrator-level programs and provides you with hands-on lab exercises. You also have the opportunity to examine static and dynamic routing through a series of labs.

Prerequisites

You should be familiar with networking concepts and have system management level experience with OpenVMS.

Course Conventions

In system displays, examples, and screen captures:

- **Courier New** identifies system display.
- **Courier New Bold** identifies user input.

Course Outline

- 1. Network Overview**
 - Network Concepts
 - IP Addresses
 - Subnet Masks
- 2. TCP/IP Commands**
 - User Services
 - PING
 - SHOW HOST
 - TELNET
 - RCP
 - RLOGIN
 - RSH
 - FINGER
 - FTP
- 3. Installing and Configuring TCP/IP Services**
 - Installation
 - Configuration
- 4. Protocols**
 - Data Link Protocol
 - IP Header Fields
 - TCP Header Fields
 - ICMP and ARP
 - UDP

5. Network File System (NFS)

- NFS Concepts
- NFS Implementation
- NFS Server File Systems
- Container File System
- NFS Management Commands
- Proxy Accounts
- Client Configuration
- Client Data Manipulation
- Server Configuration
- Server Data Manipulation

6. Serial Lines

- Serial Connections
- SLIP Encapsulation
- PPP Encapsulation
- Setting Up SLIP

7. Routing

- Routing Introduction
- Routing Table
- Static Routing
- Dynamic Routing
- ROUTED
- GATED
- Configuring Dynamic Routing

8. DNS and BIND

- History
- Concepts
- Resolver
- Server
- NSLOOKUP
- Load Broker and Metric Server

9. Bootstrap Protocol (BOOTP)

- BOOTP Concepts
- BOOTP Server
- BOOTP Client
- Server Configuration
- Server Data Files
- TFTP

10. Dynamic Host Configuration Protocol (DHCP)

- DHCP Concepts
- Advantages of Using DHCP
- DHCP Configuration Files
- Using the DHCP GUI
- DHCP Cluster Failover

11. Network Time Protocol (NTP)

- NTP Concepts
- NTP Internet Time Sources
- NTP Configuration Options
- NTP Support Files
- NTP Commands
- Trace of NTP Communication

12. Simple Network Management Protocol (SNMP)

- SNMP Concepts
- Log Files and Command Procedures
- SNMP Configuration
- Using SNMP Utilities
- Tracing SNMP Protocol Interactions

13. Remote Printing

- Remote Printing Options
- LPR/LPD Local Configuration
- LPR/LPD Remote Configuration
- Tracing Print Packets
- TELNETSYM Overview
- PC-NFS Overview

14. Troubleshooting

- Troubleshooting Options
- Configuration Troubleshooting
- Network Troubleshooting
- Performance Tuning

15. Simple Mail Transfer Protocol (SMTP)

- SMTP Concepts
- Configuring SMTP
- Using Telnet to Test SMTP
- POP and MIME Concepts

Introduction

This module describes the nature of networks and TCP/IP protocol software. It examines the protocol stack in detail. Data travels from one machine to another; this module describes how that happens.

Objectives

Upon successful completion of this module, you should be able to:

- Describe the protocol stack.
- Choose IP addresses and subnet masks.
- Describe data transfer over a network.

Topics

- Network Concepts
- IP Addresses
- Subnet Masks

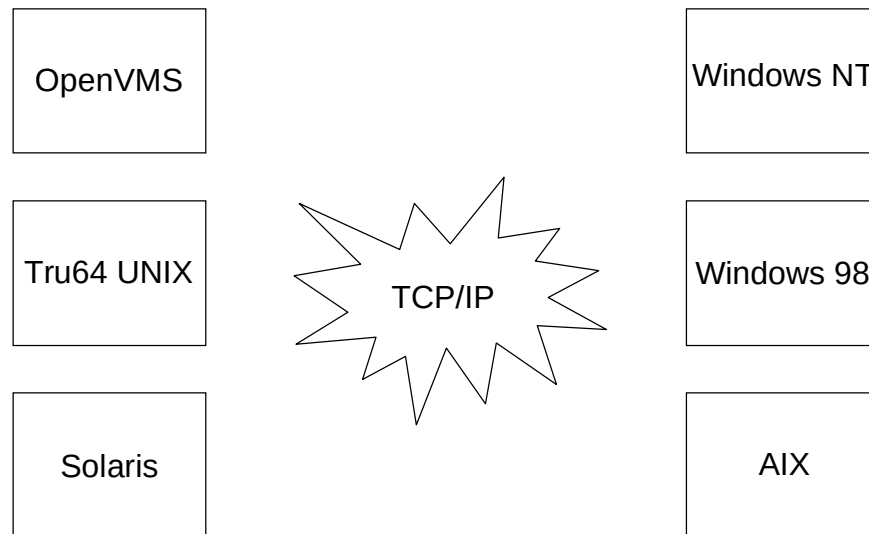
Compaq TCP/IP Services for OpenVMS

Compaq TCP/IP Services for OpenVMS provides interoperability and resource sharing among systems running OpenVMS, UNIX, Windows NT, and other operating systems that support TCP/IP.

TCP/IP Services for OpenVMS Version 5:

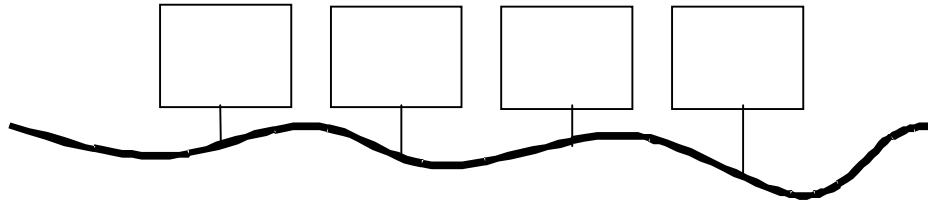
- Provides a new protocol stack based on the Compaq Tru64 UNIX TCP/IP stack.
- Is proven in the industry to be reliable, high quality, and competitive.
- Ensures transparent backward compatibility with earlier versions of TCP/IP Services for OpenVMS and runs on both OpenVMS Versions 7.2 and 7.1.

TCP/IP provides network services between heterogeneous computers and operating systems. TCP/IP concepts are the same no matter what the operating system or platform.



Network Concepts

- A network is an interconnected collection of independent computers.
- A network that covers an area such as a room, building, or campus is called a local area network (LAN).
- A LAN is a data communications system that allows independent devices to communicate directly with each other, within a moderately sized geographic area over a physical communications channel of moderate data rates (IEEE definition).
- A network that covers larger areas, such as cities or countries, is called a wide area network (WAN) or long-haul network.
- There are two basic methods for sending network messages:
 - Broadcasting (such as Ethernet and FDDI)



- Point-to-point (such as slip, ppp, and cslip)

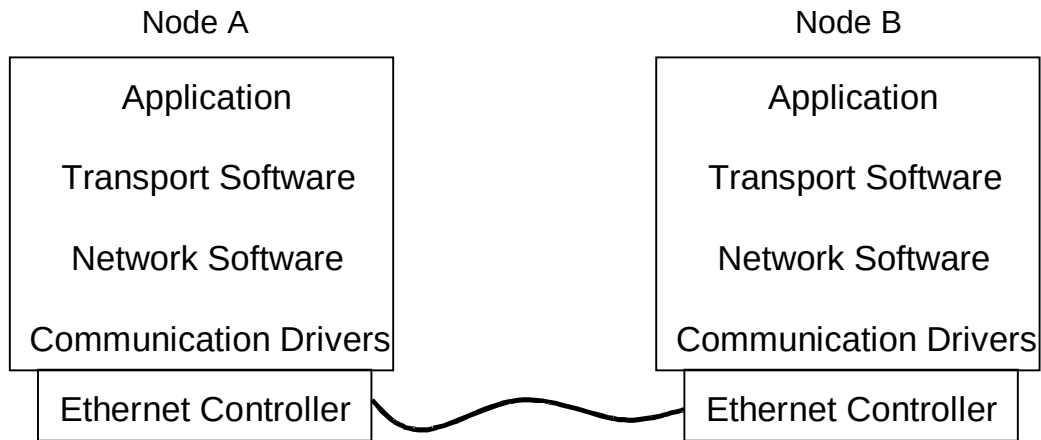


Network Communication Essentials

You need the following to create a network:

- Two or more computers connected through media
- Hardware to send and receive data
- Software to interpret and react to data

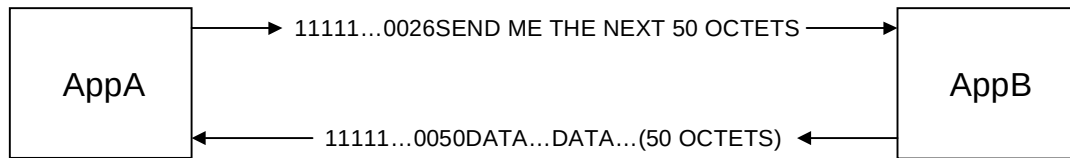
The following figure shows two nodes on a broadcast-style network (Ethernet) showing typical software configured to support network communications.



What Is a Protocol?

Two applications (AppA and AppB) need to communicate, but they are on different computers.

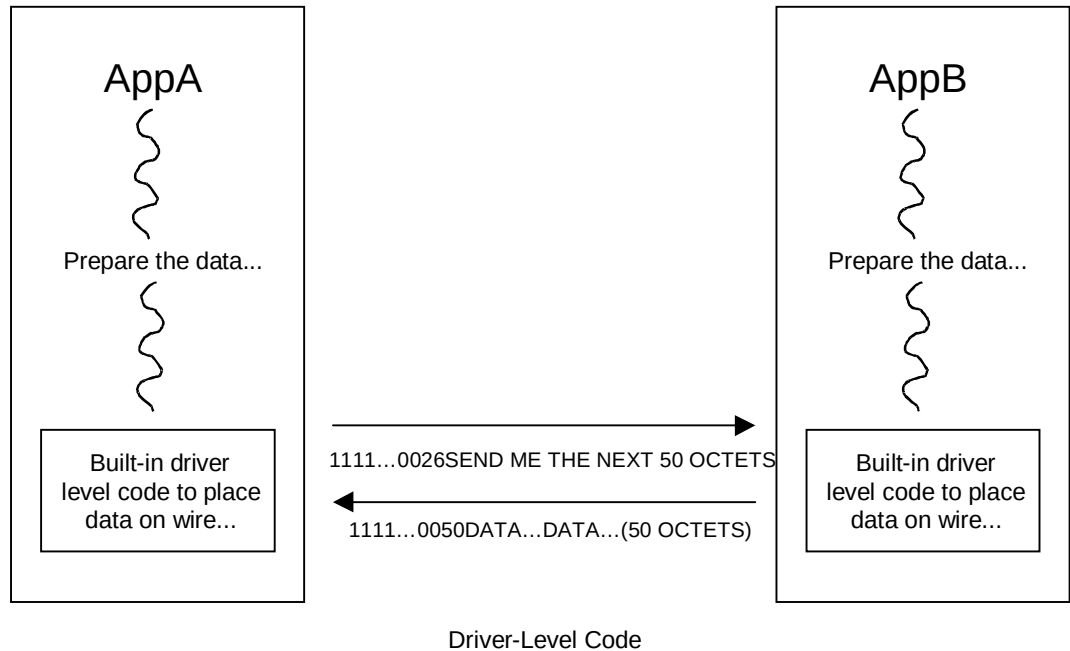
- Computers would have to be connected with wire from serial line hardware ports (or more sophisticated wire interfaces).
- Software application developers would build a protocol to formalize the interactions between the applications. This could be as simple as:
 - 4 bytes of “on” bits will be followed by 4 bytes of size, followed by the data.
- A protocol is a set of negotiated rules for communication agreed to by the software components.



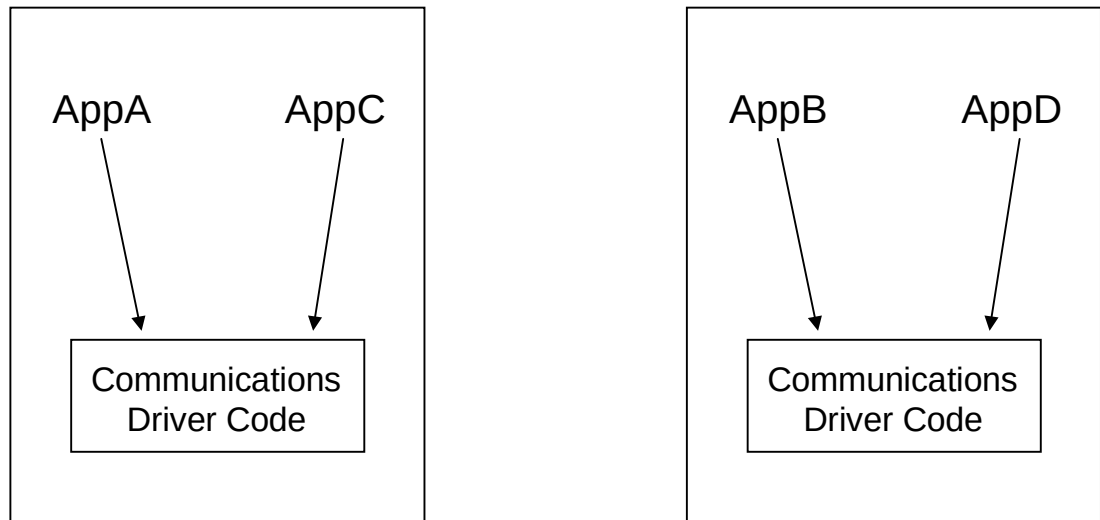
Simplistic Protocol

Physical Access to the Wire

- Applications could include specialized driver-level code to place data on the communications wire.
- What happens if more nodes are added, or the communications wire medium changes, or the driver needs to be updated?
- The following figure shows two network applications containing driver-level code.



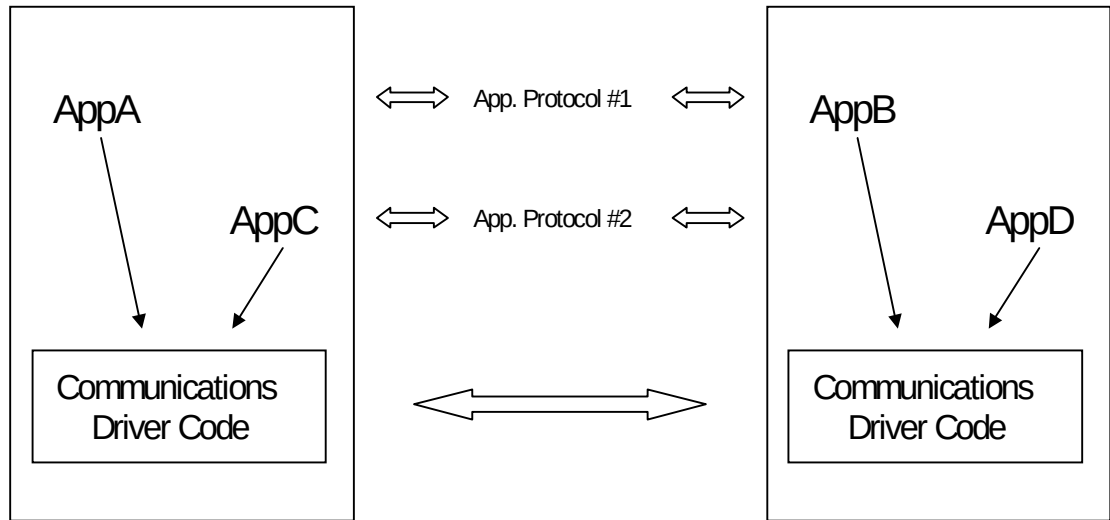
- Eventually AppC and AppD would be created with the same communication needs as AppA and AppB.
- Including driver-level code in each application that will use the wire connection between the two computers seems wasteful.
 - Build the driver code into the kernel and allow the applications to call the driver code when (and if) they need to use the wire.
 - Refer to this level code as *data link* or *physical access level*. It is implemented through the Ethernet driver or other appropriate driver-level code.
- The following figure shows applications calling communications driver-level code when needed.



Calling Driver-Level Code

Application Layer Protocol

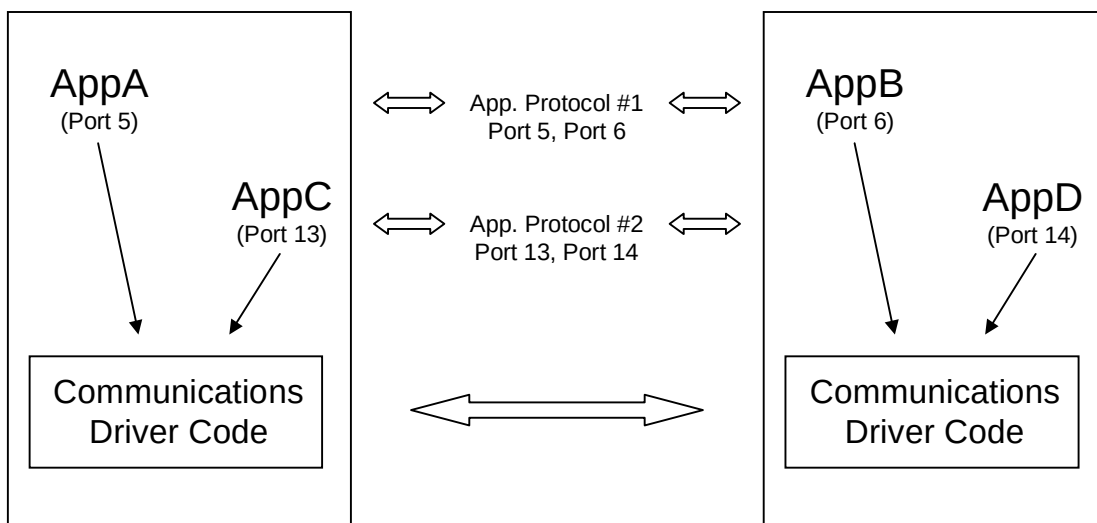
- AppC and AppD have different communication needs; therefore, they develop a different set of rules to govern their communication.
 - Any other applications could develop their own set of rules as well.
 - The rules function as a code of communications etiquette.
 - Therefore, refer to these rules of communication as *communication protocols*.
- Applications decide upon their own *application-level protocol*. They define the *application protocol level* of their software.
- The application level calls the lower level, presenting information to be placed on the wire.



Multiple Application-Level Protocols

Port Numbers

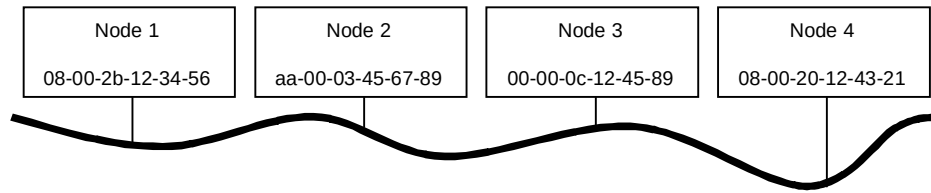
- With two computers connected by one wire, the destination computer is easy to figure.
 - But with four applications, how can the data link level software distinguish the four applications from one another?
 - The software could give each application a number.
- Think of this number as a *port number*.
- Port numbers are used to identify applications within a computer.
- Port numbers must be placed on the wire as part of the transmitted data.



Using Port Numbers to Identify Applications

Media Access Control (MAC) Addresses

- If you introduce two more machines into the picture and connect them all to a bus-based wire, the need for node addresses becomes clear.
 - Bus-based wiring (such as Ethernet) can have multiple devices connected and sharing the medium through a broadcasting technique.
 - Point-to-point connections (such as those using a serial line) involve two computers.
- Each Ethernet card (which attaches to the Ethernet wire) has a unique 48-bit media access control (MAC) address (for example, 08-00-2b-12-34-56).



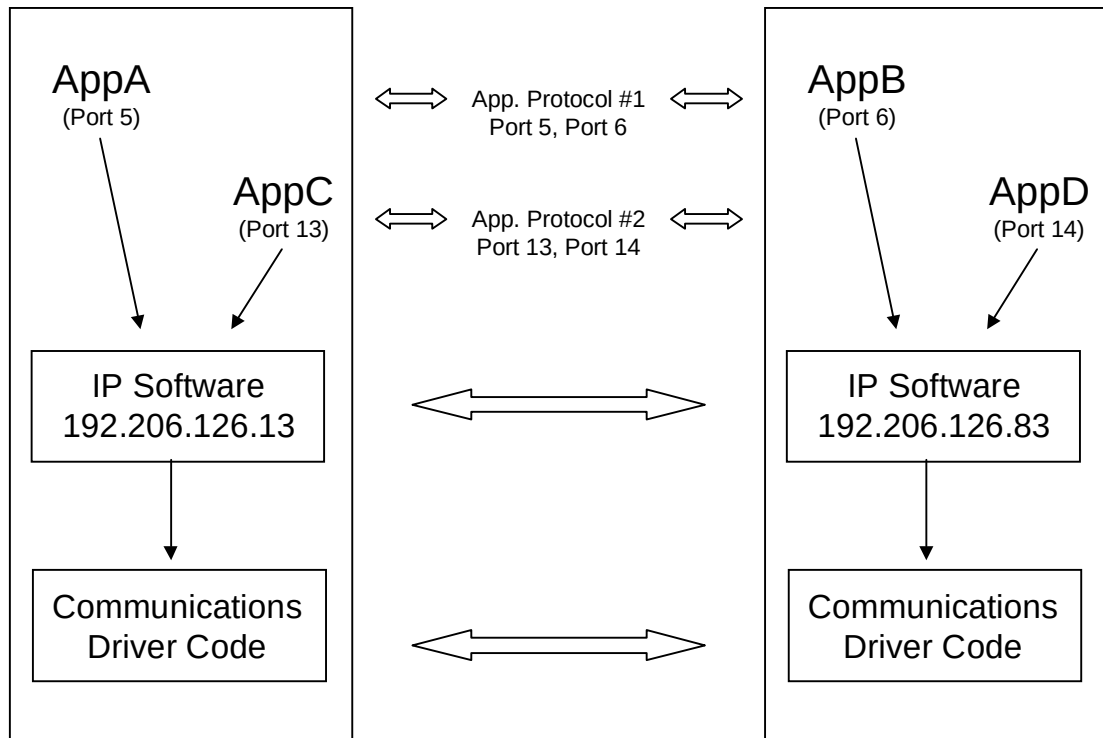
Four Nodes on an Ethernet Showing MAC Addresses

- Hardware must uniquely identify itself using a media access control address (MAC address—08:00:2b:12:34:56).
- The first 24 bits of MAC addresses are assigned by IEEE:
 - 00:00:0C — Cisco
 - 00:00:1D — Cabletron
 - 00:00:C0 — Western Digital
 - 00:AA:00 — Intel
 - 02:60:8C — 3Com
 - 08:00:2B — Digital/Compaq
 - 08:00:5A — IBM
- The network interface card supplies the MAC. This could include strategies such as collision detecting and token passing.
 - Connects to the cable.
 - Puts a final frame around the data.

Routing and Network Addresses

- Introducing more machines that are within other buildings, cities, states, or countries creates the need to **route** the packet of information in the direction of its destination.
- To keep the routing and related internetwork software compartmentalized, you build it into the kernel and let the application call it. The network software ultimately makes the calls to the data link driver software.
- The usefulness of the *network layer* of software becomes apparent.
 - The network layer on one computer must communicate with the network layer on another machine.
 - The networks need a set of rules (a protocol) to use.
 - This protocol functions between networks and is called the *Internet Protocol (IP)*.
 - A separate addressing scheme is necessary, which is useful beyond the scope of the locally pertinent MAC addresses.
- The network layer uses the IP protocol and IP addresses (for example, 192.206.126.34).

The following figure shows two nodes with IP addresses and IP software to handle routing in an intranet or the Internet.

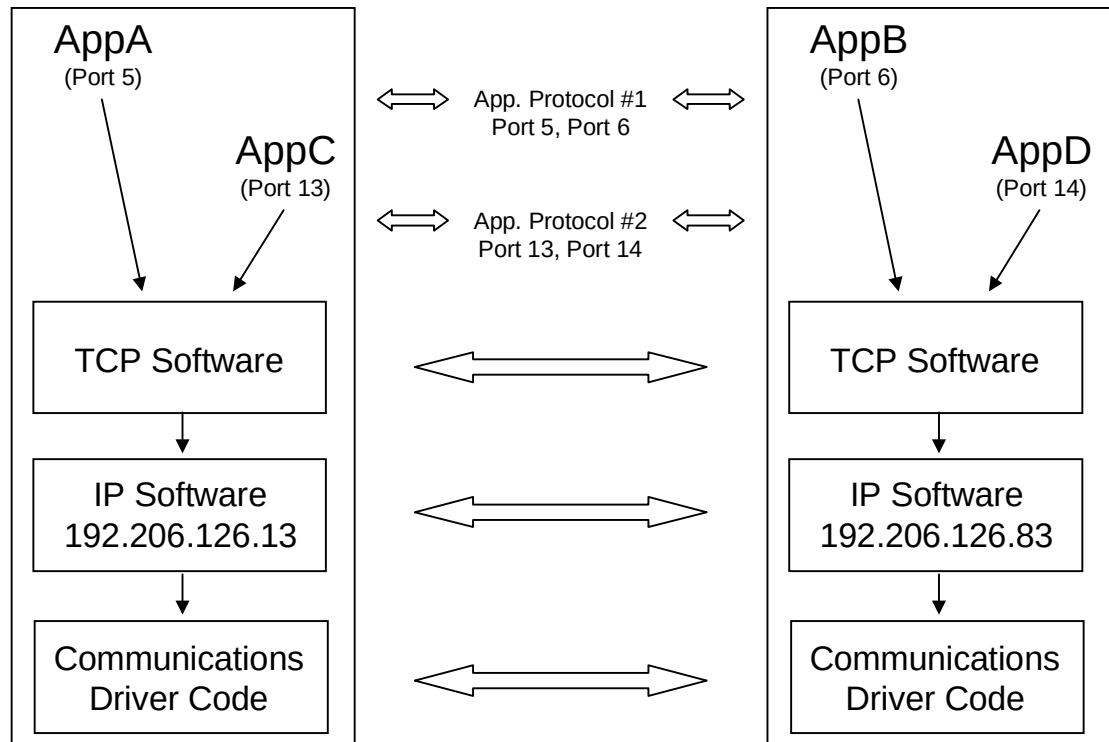


Two Nodes with IP Addresses and IP Software

Transport Layer (TCP)

- Because data traveling long distance over wires may become corrupt, the applications must contain some error correcting code. This software should be similar (if not the same) within every application.
- Break that code out of the application, build it into the kernel, and let the application call it.
- This code controls the *transmissions* made by the communicating applications. It checks for errors and locates the correct destination application through a port number.
- Because it controls the transmission of data, this protocol is called the *Transmission Control Protocol (TCP)*.
- This layer is called the *transport layer* because it provides a delivery service to the application layer (represented by a port number).

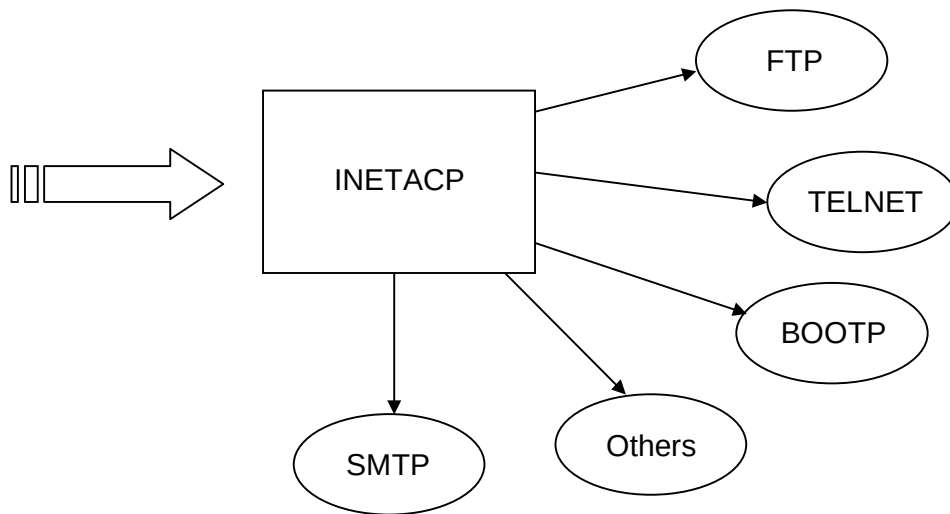
The following figure shows two nodes with modular, callable, TCP software controlling data reliability and connections.



Two Nodes with TCP Software

Multiple Services

- Many applications are common to most TCP/IP implementations.
 - They are referred to as *well-known services* or *well-known ports*.
 - Examples include ftp, telnet, rlogin, rsh, bootp, and smtp.
- Problem:
 - Network applications would clutter the machine and tie up critical resources if started at boot and allowed to function as standard server processes.
- Solution:
 - Create a single *super server* to listen for incoming requests for well-known services.
 - Have the super server (INETACP process) create the actual server processes when (and if) needed.
- The following figure shows how inbound requests for well-known services are handled by INETACP.



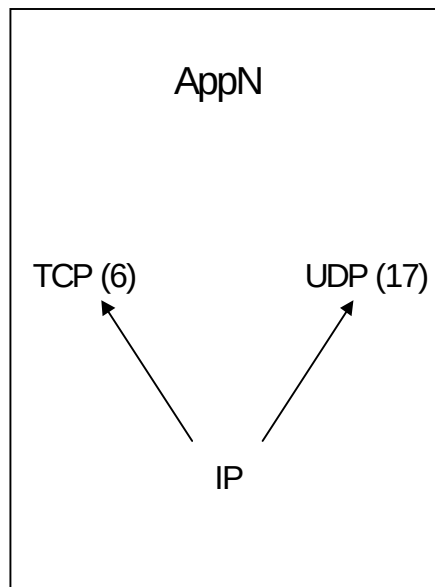
Handling Multiple Inbound Requests with INETACP

Transport Layer (UDP)

- Some applications may not be concerned about errors and may be willing to sacrifice the guarantee of accuracy for more speed.

A separate transport layer protocol relying on the user's code to make decisions about incoming data is referred to as *User Datagram Protocol (UDP)*.

- TCP must be distinguished from UDP by both the application layer and network layer.
- *Protocol numbers* are assigned to each (TCP – 6, UDP – 17).
- The following figure shows how IP software distinguishes transport level through a protocol number.

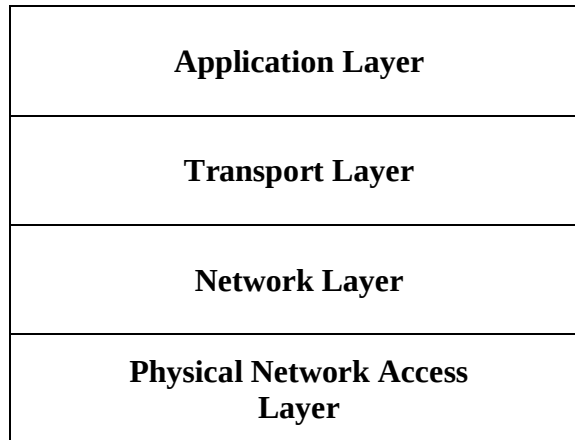


Determining Transport Level Through a Protocol Number

Protocol Stack

Rather than wastefully building driver-level software, internetwork or routing-level software, and transport-level software into every application that may want to communicate over the wires, a *protocol stack* has evolved.

- The protocol software is built and stacked one level atop the other, yielding the common reference to the protocol stack.
- The lower three levels of the stack and the associated protocols are built into the kernel and made accessible to any application that needs them.



TCP/IP Protocol Stack

Sockets

- A *socket* is an abstract representation of an end point in network communications.
 - Represents a combination of an IP address (32 bits) and a port number (16 bits).
 - Uniquely identifies an application in the Internet.
- Socket also inherently identifies, through an application-level function call (a system call in UNIX), the desired transport (TCP or UDP).
 - An application opens a socket like it opens a file.
 - Reads and writes are directed transparently through transport-, network-, and physical-level software to the application.
- Sockets are represented by a socket data structure in the memory of each participating node.

The Internet

- Internetworking accommodates multiple, diverse underlying hardware technologies and provides a way to interconnect heterogeneous networks.
- The Internet is a network of networks.
 - Designed by the Advanced Research Projects Agency (ARPA) in 1969.
 - Set of network standards specifying how computers communicate.
 - The standards reached their current form from 1977-1979.
 - Changes are driven through the RFC (Request For Comment) process.
 - Conversion to TCP/IP started in earnest in 1980.
 - 4.2 BSD UNIX (1982) contained TCP/IP.
- Universities and government contractors joined first, followed by the rest of the United States, then the world!
 - In 1987 growth was around 15% per month.
 - In 1994 Internet included over 3 million computers in 61 countries.
 - Roughly doubled in size each year until recently.
 - Growth is monitored and closely controlled now (ICANN).
- The Internet uses the TCP/IP protocols for communication.
 - TCP, UDP, and IP protocols (among many others) contain the details of message formats and responses.
 - TCP/IP is independent of hardware.

IP Addresses

- To receive and deliver packets successfully between hosts, IP relies on two pieces of information that the user provides:
 - IP address (identifies a computer as a 32-bit address that is unique across a TCP/IP network)
 - Subnet mask
- IP address is usually represented in dotted decimal notation, which depicts each octet (8 bits, or 1 byte) of the IP address as its decimal value and separates each octet with a period.
 - 102.54.94.97
- It is usually associated with a name found in a local file (such as `alfman`, `snoopy`, or `bigguy`).
 - UNIX file is `/etc/hosts`.
 - OpenVMS file is pointed to by logical name `TCPIP$HOSTS`.
- Node name translations can also be served to clients through BIND software.
 - BIND is Berkeley Internet Name Domain, the most commonly used name server software.
 - Examples include: **`alfman.bruden.com`**, or **`snoopy.alf.dec.com`**, or **`bigguy.bldg3.west2.bigcomp.org`**.
 - These node names are referred to as “canonical” names (names that have been “canonized”). They are the official names for these nodes.



Important

Names are always translated into IP addresses!

Network ID and Host ID

- IP address (32 bits) contains two pieces of information:
 - Network ID
 - Host (or system) ID for your computer.
- Network ID identifies a group of computers and other devices that are all located on the same logical network.

In internetworks (networks formed by a collection of local area networks), there is a unique network ID for each network.
- Host ID identifies your computer within a particular network ID.

Host is any device that is attached to the network and uses IP.
- Networks that connect to the public Internet must obtain an official network ID from the Internet Corporation for Assigned Names and Numbers (ICANN), which is replacing InterNIC, to guarantee IP network ID uniqueness.

The network administrator must assign unique host IDs for computers within the local network.
- Private networks not connected to the Internet can choose to use their own network identifier.
- Obtaining a valid network ID from ICANN allows a private network to connect to the Internet in the future without reassigning addresses.

Internet Addresses

- Each address is unique to the Internet.
- You can examine the address to determine a network's class, Network ID, and host number (and a subnet number if subnets are being used.)
 - The first field of an address determines a class: A, B, or C.
(Note: Class D—Multicast and Class E—Reserved are not discussed here.)
 - Interpretation of other fields in the address depends on the class.
- The Internet community has defined address classes to accommodate networks of varying sizes.
- You can determine network class from the first octet of its IP address.
- Currently, most TCP/IP implementations use IP Version 4. IP Version 6 (IPNG—next generation) will use 128-bit addresses.

IP Address Classes

- Class A address: 16.45.67.8
 - 16 — network number.
 - 45.67.8 — host number.
 - First byte — **less than 127**.
 - If bit 31 is 0, next seven bits are network ID.
- Class B address: 128.45.67.8
 - 128.45 — network number.
 - 67.8 — host number.
 - First byte ranges between **128 and 191 (inclusive)**.
 - If bits 31 and 30 are 1,0, next 14 bits are network ID.
- Class C address: 193.45.67.8
 - 193.45.67 — network number.
 - 8 — host number.
 - First byte ranges between **192 and 223 (inclusive)**.
 - If bits 31 to 29 are 1,1,0, next 21 bits are network ID.
- Class D Address
 - Bytes 224 to 239 are for multicast uses.
 - If bits 31 to 29 are 1,1,1 message is multicast.
- Class E Address
 - First byte greater than 239 — reserved for future use and testing.

Subnet Masks

- Because the sender's IP address is included in every outgoing IP packet, it is useful for the receiving computer system to derive the originating network ID and host ID from the IP address field.
- Achieve this by using a subnet mask (sometimes called a *netmask*).
 - 32-bit values that allow the recipient of IP packets to distinguish the network ID portion of the IP address from the host ID.
 - Value of a subnet mask is represented in dotted decimal notation or sometimes shown in hex (**ifconfig** does this).
 - So, 255.255.255.0 is FF FF FF 00 in hex.
 - ♦ 255.255.255.224 is FF FF FF E0.
 - ♦ (E in hex is 1110 in binary.)
 - ♦ (F in hex is 1111 in binary.)
- Determine the subnet mask by assigning ones to bits that belong to the network ID and zeros to the bits that belong to the host ID.
- The following table shows default subnet masks for standard IP address classes.

Address Class	Bits for Subnet Mask	Subnet Mask
Class A	11111111 00000000 00000000 00000000	255.0.0.0
Class B	11111111 11111111 00000000 00000000	255.255.0.0
Class C	11111111 11111111 11111111 00000000	255.255.255.0

- The result allows IP to determine the host and network IDs of the local computer.
- When the IP address is 132.54.94.97 and the subnet mask is 255.255.0.0, the network ID is 132.54 and the host ID is 94.97.

- All IP implementations ask for a subnet mask during configuration. The mask can be used by software (IP) to “mask” the bits that are devoted to identifying the network ID as opposed to the node ID.
- If network ID bits are masked, the remainder is the node number.
- If node bits are masked, the remainder is the network number.
- Consider the standard 32-bit IP address to be broken into two parts:
 - Network ID
 - Node ID
- The line of demarcation between the network bits and node bits is determined by the subnet mask (assume contiguous mask bits for now).

ICANN (formerly NIC) assigns network IDs, not node IDs.

Class C network ID example:	192	.206	.126	.23
Default Class C subnet mask:	255	.255	.255	.0
In binary:	11111111	.11111111	.11111111	.00000000

- The line of demarcation between the network bits and the node bits in an IP address is the point where the ones become zeros in the subnet mask.

Nondefault Subnet Masks

- You can use subnet masks to further segment an assigned network ID among several local networks.
 - For example, suppose a network is assigned the Class B network address 135.100.
 - This is one of the more than 16,000 Class B addresses capable of serving more than 65,000 nodes per network.
 - However, suppose the worldwide corporate network to which this ID is assigned is composed of 12 international LANs with 75 to 100 nodes each.
- Instead of applying for 11 more network IDs, it is better to use subnetting to make more effective use of the assigned ID 135.100.
 - The third octet of the IP address can be used as a subnet ID, to define the subnet mask 255.255.255.0.
 - This splits the Class B address into 256 subnets, 135.100.0 through 135.100.255, each of which can have 254 nodes. (Host IDs 0 and 255 should not be assigned to a computer; they are used as broadcast addresses, which are typically recognized by all IP implementations.)
 - Any 12 of these network addresses could be assigned to the international LANs in this example.
- Within each LAN, each computer is assigned a unique host ID, and they all have the subnet mask 255.255.255.0.
- Sometimes it is necessary to segment only portions of an octet, using only a few bits to specify subnet IDs (such as when Class B subnets exceed 256 nodes, or with Class C subnetting).
- For all systems on the local network, the subnet mask must be the same for that network ID.

Class C Subnet Case

- Given the following:
 - Assigned Class C network ID is 192.206.126
 - Company has five LANs with no more than 20 nodes on each.
 - What would be a reasonable subnet mask?
- Consider the following:

255.255.255.128 (11111111.11111111.11111111.10000000)

 - Allows two subnets (0 and 1), with each subnet able to have 126 nodes ($2^7 - 2$ or $128 - 2$).

255.255.255.192 (11111111.11111111.11111111.11000000)

 - Allows four subnets (0, 1, 2, 3), with each subnet able to have 62 nodes ($2^6 - 2$ or $64 - 2$).

255.255.255.224 (11111111.11111111.11111111.11100000)

 - Allows 8 subnets (0 through 7), with each subnet able to have 30 nodes ($2^5 - 2$ or $32 - 2$).
- And:

255.255.255.240	16 subnets	14 nodes each
255.255.255.248	32 subnets	6 nodes each
255.255.255.252	64 subnets	2 nodes each
255.255.255.254	128 subnets	0 nodes each
255.255.255.255	256 subnets	0 nodes each

- This example demands five subnets, so the choice is 255.255.255.224, which provides the five needed plus an additional three for growth. (This choice is limited to 30 nodes maximum in each subnet.)
- For this case, choose subnet numbers 1 through 5.

**Important**

Subnet 0 is available for use, although some older routers have trouble with it. The original RFCs implied that you should not use subnet numbers of (binary) all zeros or (binary) all ones. However RFC 1812, *Requirements for IP Version 4 Routers*, makes it clear that subnets of all zeros or all ones are legal and should be supported by all routers.

(*TCP/IP Network Administration*, Second Edition, Craig Hunt, O'Reilly, pg. 33.)

- What IP address would be used for node 9 in subnet 3?
 - True subnetting breaks up the IP address into three components:
 - ♦ Network ID
 - ♦ Subnet number
 - ♦ Node number
- In this case, the network ID is 192.206.126, the subnet number is 3, the node number is 9.

Net ID	Subnet	Node	
192.206.126.	---	-----	
255.255.255.	1 1 1	0 0 0 0 0	(mask of 255.255.255.224)

- Using the lines to break up the 8 bits, the following represents a 3 in the subnet section (011) and a 9 in the node section (01001). This yields 0 1 1 0 1 0 0 1 when the two binary values are concatenated into an 8-bit entity.

Net ID	Subnet	Node	
	3	9	
192.206.126.	0 1 1	0 1 0 0 1	
255.255.255.	1 1 1	0 0 0 0 0	(mask of 255.255.255.224)

- Using the templates, notice that 0 1 1 0 1 0 0 1 equals decimal 105 (64+32+8+1).
- Therefore, the final IP address is **192.206.126.105**.

- Furthermore, using the same analysis:

Node 26 in subnet 2 has IP address 192.206.126.0 1 0 1 1 0 1 0, 192.206.126.90

Node 13 in subnet 5 has IP address 192.206.126.1 0 1 0 1 1 0 1, 192.206.126.173

Node 3 in subnet 1 has IP address 192.206.126.0 0 1 0 0 0 1 1, 192.206.126.35

Node 30 in subnet 3 has IP address 192.206.126.0 1 1 1 1 1 1 0, 192.206.126.126

- Given an IP address of 192.206.126.187, and a subnet mask of 255.255.255.224, what subnet is this node on, and what is the node number?
- Using the templates (or a handy calculator), 187 is 10111011 in binary.

! **Important**

• Analyze the bits between the solid network ID line and the dotted subnet line as if they were standing alone. That is, the 101 in the next example is interpreted as one group of ones and one group of fours, not one group of 32s and one group of 128s.

- Applying the mask as shown earlier yields:

		5	27	
		/	\	
192.206.126		<u>1 0 1</u>	<u>1 1 0 1 1</u>	(192.206.126.187)
255.255.255		<u>1 1 1</u>	<u>0 0 0 0 0</u>	(mask of 255.255.255.224)

- Therefore, the subnet number (101 binary) is 5, and the node number (11011) is 27.

■ Likewise:

- 192.206.126.47 with mask of 255.255.255.192 is node 47 in subnet 0.

192.206.126	0 0 1 1 1 0 1 1	(192.206.126.47)
255.255.255.	1 1 0 0 0 0 0 0	(mask of 255.255.255.192)

- 192.206.126.157 with mask of 255.255.255.240 is node 13 in subnet 9.

192.206.126	1 0 0 1 1 1 0 1	(192.206.126.157)
255.255.255	1 1 1 1 0 0 0 0	(mask of 255.255.255.240)

- 192.206.126.157 with mask of 255.255.255.128 is node 29 in subnet 1.

192.206.126	1 0 0 1 1 1 0 1	(192.206.126.157)
255.255.255	1 0 0 0 0 0 0 0	(mask of 255.255.255.128)

Broadcast Addresses — Class C Subnet Case

- Broadcast address (all node bits on) for each of the five subnets would be:

Subnet #1 - 192.206.126.	<u>0 0 1</u>	<u>1 1 1 1 1</u>	, 192.206.126. 63
Subnet #2 - 192.206.126.	<u>0 1 0</u>	<u>1 1 1 1 1</u>	, 192.206.126. 95
Subnet #3 - 192.206.126.	<u>0 1 1</u>	<u>1 1 1 1 1</u>	, 192.206.126. 127
Subnet #4 - 192.206.126.	<u>1 0 0</u>	<u>1 1 1 1 1</u>	, 192.206.126. 159
Subnet #5 - 192.206.126.	<u>1 0 1</u>	<u>1 1 1 1 1</u>	, 192.206.126. 191

- Broadcast addresses for the subnets not currently in use would be:

Subnet #0 -192.206.126.	<u>0 0 0</u>	<u>1 1 1 1 1</u>	, 192.206.126. 31
Subnet #6 -192.206.126.	<u>1 1 0</u>	<u>1 1 1 1 1</u>	, 192.206.126. 223
Subnet #7 -192.206.126.	<u>1 1 1</u>	<u>1 1 1 1 1</u>	, 192.206.126. 255

Introduction

Compaq TCP/IP Services for OpenVMS provides commands that are familiar to a UCX user as well as commands that are UNIX based. This module provides a brief look at some of the categories and commands that the software provides. This course covers many of these categories in more detail in later modules.

Objectives

Upon completion of this module, you should be able to:

- Use TCP/IP Services for OpenVMS standard commands.
- Use UNIX style commands.

Topics

- User Services
- PING
- SHOW HOST
- TELNET
- RCP
- RLOGIN
- RSH
- FINGER
- FTP

User Services

- TCP/IP includes application services providing file transfer, remote logins, file sharing, mail, and other network services.
- Most utilities can reference a remote node by IP address or node name. Node name is available, if found, in the TCPIP\$HOST file or resolvable through BIND.
- PING provides basic connectivity and protocol checking (up to the network level).
- SHOW HOST displays entries in the TCPIP\$HOST file. It can also display BIND served host names.

```
$ tcpip show host alfman
```

```
LOCAL database
```

Host address	Host name
192.206.126.3	ALFMAN

```
SHOW HOST displaying local TCPIP$HOST database.
```

```
$
```

```
$
```

```
$ tcpip show host
```

```
LOCAL database
```

Host address	Host name
192.206.126.3	ALFMAN
127.0.0.1	LOCALHOST, localhost
192.206.126.27	den255, DEN255

PING using IP address.

\$

\$ tcpip ping 192.206.126.80

PING 192.206.126.80 (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=128 time=2 ms

(...)

----192.206.126.80 PING Statistics----

6 packets transmitted,

6 packets received,

0% packet loss

round-trip (ms) min/avg/max = 1/1/2 ms

\$

\$ tcpip ping compcom

ping: unknown host compcom

\$

PING using host name — unknown host.

\$ tcpip set host compcom /address=192.206.126.80

SET HOST adding to local TCPIP\$HOSTS database.

\$ tcpip show host

LOCAL database

Host address	Host name
192.206.126.3	ALFMAN
192.206.126.80	COMPCOM
127.0.0.1	LOCALHOST, localhost
192.206.126.27	den255, DEN255

\$

\$ **tcpip ping compcom**

PING COMPCOM (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=128 time=1 ms

(...)

PING using host name.

----COMPCOM PING Statistics----

5 packets transmitted,

5 packets received,

0% packet loss

round-trip (ms) min/avg/max = 1/1/1 ms

\$

Remote Computing with Telnet

- Log in to a remote host using session customization options.
- Establish multiple connections and toggle between them.

\$ telnet alfman

Connecting to the UNIX system with the telnet command.

%TELNET-I-TRYING, Trying ... 192.206.126.3

%TELNET-I-SESSION, Session 01, host alfman, port 23

Remote Telnet server found using port 23.

-TELNET-I-ESCAPE, Escape character is ^]

Digital UNIX (alfman) (ttyp4)

login: **sloccer**

Password:

Digital UNIX V4.0E (Rev. 1091);

(...)

alfman> **whoami**

sloccer

alfman>

alfman> **exit**

%TELNET-S-REMCLOSED, Remote connection closed

-TELNET-I-SESSION, Session 01, host alfman, port 23

\$

Proxy Accounts

Use proxy accounts for NFS server, NFS client, PC-NFS, RSH, LPR/LPD, and user applications. The proxy database (TCPIP\$PROXY) holds the proxy accounts.

There are two types of proxies:

- **Communication proxy**

- Provides an identity for users of RSH, RLOGIN, REXEC, RMT/RCD, LPR/LPD, and user applications (if these applications are marked with SET SERVICE /FLAGS=APPLICATION_PROXY).
- Required qualifiers: /HOST, /REMOTE_USER.
- Establishes a form of “trust” between nodes.
- Format for communication proxies:

```
ADD PROXY user { /HOST=host | /REMOTE_USER=user }  
               [ /PERMANENT ]  
(... others)
```

- **NFS Proxy** (discussed in the NFS module)

Remote Computing with RCP

- Copy files between the local host and a remote host or between two remote hosts.
- Request authentication occurs on the remote host using the user name supplied by RCP. The proxy database is checked to see if a proxy identity has been established for the user.

```
$ rcp "alfman:/etc/hosts.equiv" [ ]
```

RCP copying from UNIX to OpenVMS.

This RCP command specifies no user name, so the current user (system) is used to access alfman (assuming no proxy).

```
$
```

```
$ dir host*
```

```
Directory SYS$SYSROOT:[SYSMGR]
```

```
HOSTS.EQUIV;1
```

```
Total of 1 file.
```

```
$
```

```
$
```

```
$ type hosts.equiv
```

View contents of the file copied from Tru64 UNIX system alfman.

```
(...)
```

```
#      host1 [login_n]
```

```
#
```

```
# host1      name of a host considered trusted by the local system
```

```
# [login_n]  individual user on the trusted host that can log in
```

```
#      without a password; if no users are specified then all
```

```
#      users on the trusted host can log in without a password
```

```
#
```

```
# A single plus sign (+) on a line by itself signifies "match any".
```

```
# "Match any" implies *all* hosts are considered trusted. Because
```

```
# of the security ramifications of this uncommenting the NO_PLUS
```

```
# keyword in the following line, globally turns off "match any".
```

```
#NO_PLUS
```

```
den255 system
```

System alfman trusts user named system on den255.

\$

\$ **rcp /username=obrien "alfman:/etc/hosts.equiv" []**

RCP command specifying a different (untrusted) user name.

Permission denied.

%RCP-E-CONHST, error connecting to remote host ALFMAN

Permission denied because of lack of trust—no mention of user obrien in the hosts.equiv file.

\$

\$ **rcp /username=obrien/password="Bruden" "alfman:/etc/hosts.equiv" []**

RCP command specifying user name and password.

\$

\$ **dir host***

Directory SYS\$SYSDIR: [SYSDIR]

HOSTS.EQUIV;2 HOSTS.EQUIV;1

Total of 2 files.

Remote Computing with RLOGIN

- Connect to a remote host and start a login session.
- Request authentication occurs on the remote host using the user name supplied by RLOGIN.

\$ rlogin alfman

Password:

Logging in to the system account on node alfman.

Digital UNIX V4.0E (Rev. 1091);

(...)

alfman> **whoami**

system

alfman> **exit**

%RLOGIN-S-REMCLOSED, Remote connection closed

\$

\$ rlogin /username=obrien alfman

Password:

Logging in to the obrien account on node alfman.

Last login: Wed Jun 9 12:00:28 from den255

Digital UNIX V4.0E (Rev. 1091);

(...)

alfman> **cat /etc/hosts.equiv**

(...)

host1 [login_n]

#

host1 name of a host considered trusted by the local system

[login_n] individual user on the trusted host that can log in

without a password; if no users are specified then all

users on the trusted host can log in without a password

(...)

alfman> **exit**

%RLOGIN-S-REMCLOSED, Remote connection closed

\$ rlogin alfman

Last login: Wed Jun 9 12:02:00 from den255

Digital UNIX V4.0E (Rev. 1091);

(...)

Logging in to the system account on node alfman using trust through
/etc/hosts.equiv.

alfman> **tail /etc/hosts.equiv**

[login_n] individual user on the trusted host that can log in

without a password; if no users are specified then all

users on the trusted host can log in without a password

(...)

den255

Node alfman trusts node den255.

alfman> **exit**

%RLOGIN-S-REMCLOSED, Remote connection closed

\$

Remote Computing with RSH

- Connect to a remote host and execute a command.
- Request authentication occurs on the remote host using the user name supplied to RSH.

```
$ rsh alfman tail /etc/hosts.equiv
```

Executing the `tail` command on UNIX node `alfman`.

```
# [login_n] individual user on the trusted host that can log in
# without a password; if no users are specified then all
# users on the trusted host can log in without a password
#
# A single plus sign (+) on a line by itself signifies "match any".
# "Match any" implies *all* hosts are considered trusted. Because
# of the security ramifications of this uncommenting the NO_PLUS
# keyword in the following line, globally turns off "match any".
#NO_PLUS
den255
```

```
$
```

The RSH/PASSWORD command:

- Uses the REXEC facility to connect to the remote host, which executes the command you specify.
- Causes request authentication to occur on the remote host using the user name and password supplied by RSH.

Remote Computing with FINGER

Display information about users logged in to a remote host, such as their login user names or programs they are using.

```
$ finger @alfman
```

Showing users on node alfman.

```
[ALFMAN]
```

Login	Name	TTY	Idle	When	Office
root	system PRIVILEGED	ac	co	5:19 Wed 06:51	
root	system PRIVILEGED	ac	*:0	Wed 06:51	
sweeney	D. Gus Sweeney	p1	1	Wed 12:08	
root	system PRIVILEGED	ac	p2	4:00 Wed 06:51	
root	system PRIVILEGED	ac	p3	1 Wed 06:51	
obrien	Dennis O'Brien	p4		Wed 12:09	
sloccer	Dave E. Sloccer	p5		Wed 12:09	
lyffe	Doctor Xavier Lyffe	p6		Wed 12:08	

```
$
```

```
$
```

```
$ finger lyffe@alfman
```

Showing information about user lyffe on node alfman.

```
[ALFMAN]
```

```
Login name: lyffe                      In real life: Doctor Xavier Lyffe
```

```
Directory: /usr/users/lyffe           Shell: /bin/ksh
```

```
On since Jun  9 12:08:51              1 minute 15 seconds Idle Time
```

```
on ttyp6
```

```
No Plan.
```

```
$
```

```
$
```

File Transfer with FTP

Create, delete, and copy files and directories between hosts.

```
$ ftp alfman
```

```
220 alfman FTP server (Digital UNIX Version 5.60) ready.
```

```
Connected to ALFMAN.
```

```
Name (ALFMAN:system):obrien
```

```
331 Password required for obrien.
```

```
Password:
```

```
230 User obrien logged in.
```

```
FTP>
```

```
FTP>
```

```
FTP> pwd
```

```
257 "/usr/users/obrien" is current directory.
```

Shows the current working directory on the remote node.

```
FTP>
```

```
FTP>
```

```
FTP> show def
```

```
257 "/usr/users/obrien" is current directory.
```

Also shows the current working directory on the remote node.

```
FTP>
```

```
FTP>
```

```
FTP> lcd sys$scratch
```

```
Local directory now SYS$SYSROOT:[SYSMGR]
```

Sets the local directory.

```
FTP>
```

```
FTP>
```

```
FTP> set def "/etc"
```

```
250 CWD command successful.
```

Sets the remote directory.

FTP>

FTP>

FTP> **pwd**

257 "/etc" is current directory.

FTP>

FTP>

FTP> **dir**

200 PORT command successful.

150 Opening ASCII mode data connection for /bin/ls (192.206.126.27,1038).

total 1112

-rwxr-xr-x 1 bin bin 2737 Nov 13 1998 #.mrg..binlog.conf

-rwxr-xr-x 1 bin bin 30514 Nov 13 1998 #.mrg..ddr.dbase

(...)

drwxr-xr-x 10 root system 1536 Feb 15 18:13 zoneinfo

226 Transfer complete.

15165 bytes received in 00:00:00.31 seconds (47.46 Kbytes/s)

FTP>

FTP>

FTP> **get /etc/hosts**

200 PORT command successful.

150 Opening ASCII mode data connection for /etc/hosts (192.206.126.27,1039)
(1995 bytes).

226 Transfer complete.

local: SYS\$SYSROOT:[SYSMGR]_ETC_HOSTS.;1 remote: /etc/hosts

1995 bytes received in 00:00:00.02 seconds (92.77 Kbytes/s)

Transfers /etc/hosts from the remote node.

FTP>

FTP>

FTP>

FTP> **bye**

221 Goodbye.

\$

\$

\$ **type sys\$scratch:_etc_hosts.;1**

(...)

127.0.0.1 localhost

192.206.126.3 alfman alfman #HOSTNAME for Interface ln0

192.206.126.27 den255

\$

\$

Note the alteration of the file specification.

File Transfer with TFTP

- Download and transfer files.
- Not typically used at the application level.

Resource Sharing

Service	Description
LPD/LPR	Print files on remote and local hosts.
TELNETSYM	Print files on remote hosts using the Telnet protocol.
NFS	Authenticate requests and provide access to remote files.
Mail	
SMTP	Send and receive electronic mail from remote hosts.
POP	Send and receive electronic mail from your PC.

Network Services

Service	Description
BIND	Name and address resolution service distributing host name information (for example, alfman.den.bruden.com).
SNMP	Monitor and manage network devices from across an internetwork.
NTP	Synchronize time between hosts.
BOOTP	Provide for remote boot requests.
DHCP	Dynamically configure and maintain your IP address space.
SLIP	Connect a node to a network over a serial line using IP.
CSLIP	Connect a node to a network over a serial line using compressed IP.
PPP	Connect a node to a network using IP or other supported network protocols over a serial line.

Management Services

Service	Description
TCPIP	Management Control Program
TCPTTRACE	Trace low-level packets and format protocol headers (like tcpdump).
NSLOOKUP	Communicate with local or remote name servers (BIND servers).
UNIX Commands	Several UNIX based commands are available within TCP/IP Services for OpenVMS V5 (such as ifconfig, sysconfig, netstat, arp, route, and ripquery).

Management Examples

Use Address Resolution Protocol (ARP) to relate IP addresses to MAC addresses.

```
TCPIP> arp -a
```

ARP command showing nothing in the ARP table.

```
TCPIP>
```

```
TCPIP> arp alfman
```

```
alfman (192.206.126.3) -- no entry
```

```
TCPIP>
```

```
TCPIP>
```

```
TCPIP> ping alfman
```

```
PING ALFMAN (192.206.126.3): 56 data bytes
```

```
64 bytes from 192.206.126.3: icmp_seq=0 ttl=64 time=3 ms
```

```
64 bytes from 192.206.126.3: icmp_seq=1 ttl=64 time=1 ms
```

```
-----ALFMAN PING Statistics-----
```

```
2 packets transmitted, 2 packets received, 0% packet loss
```

```
round-trip (ms)  min/avg/max = 1/2/3 ms
```

```
TCPIP>
```

```
TCPIP>
```

```
TCPIP> arp -a
```

```
ALFMAN (192.206.126.3) at 08-00-2b-39-80-cb
```

After a PING, the ARP table has information about address 192.206.126.3.

```
TCPIP>
```

TCPIP>

TCPIP> **ping compcom**

PING COMPCOM (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=128 time=1 ms

64 bytes from 192.206.126.80: icmp_seq=1 ttl=128 time=1 ms

----COMPCOM PING Statistics----

2 packets transmitted, 2 packets received, 0% packet loss

round-trip (ms) min/avg/max = 1/1/1 ms

TCPIP>

TCPIP>

TCPIP> **arp -a**

COMPCOM (192.206.126.80) at 00-40-d0-08-a2-b3

ALFMAN (192.206.126.3) at 08-00-2b-39-80-cb

Two items now in ARP table.

TCPIP>

TCPIP>

TCPIP> **ifconfig -a**

The ifconfig command showing all (-a) network interfaces.

TN0: flags=80<NOARP>

WE0: flags=c43<UP, BROADCAST, RUNNING, MULTICAST, SIMPLEX> Hwaddr aa-00-04-00-01-04

inet 192.206.126.27 netmask ffffffff00 broadcast 192.206.126.255 ipmtu 1500

L00: flags=100c89<UP, LOOPBACK, NOARP, MULTICAST, SIMPLEX, NOCHECKSUM>

inet 127.0.0.1 netmask ff000000 ipmtu 4096

TCPIP>

TCPIP>

TCPIP> **netstat -nr**

The netstat command showing routing table statistics.

Routing tables

Destination	Gateway	Flags	Refs	Use	Interface
-------------	---------	-------	------	-----	-----------

Route Tree for Protocol Family 2

127.0.0.1	127.0.0.1	UH	3	1520	LO0
192.206.126/24	192.206.126.27	U	2	2513	WE0

Route Tree for Protocol Family 26

TCPIP>

TCPIP>

TCPIP> **netstat -I**

The netstat command showing interface statistics.

Name	Mtu	Network	Address	Ipkts	Ierrs	Opkts	Oerrs	Coll
TN0*	1280	<Link>	Link#2	0	0	0	0	0
WE0	1500	<Link>	aa:00:04:00:01:04	927	0	923	0	0
WE0	1500	192.206.126	den255	927	0	923	0	0
LO0	4096	<Link>	Link#1	3143	0	3143	0	0
LO0	4096	127	LOCALHOST	3143	0	3143	0	0

TCPIP>

TCPIP>

TCPIP> **sysconfig -s**

The sysconfig -s command showing configured subsystems.

inet: loaded and configured
socket: loaded and configured
arp: loaded and configured

TCPIP>

TCPIP>

TCPIP> **sysconfig -q inet**

Querying the inet subsystem with sysconfig.

inet:

inifaddr_hsize = 32

ipdefttl = 64

ipdirected_broadcast = 0

ipforwarding = 0

ipfragttl = 60

ipgateway = 0

ipport_userreserved = 5000

ipsendredirects = 1

ipsrcroute = 1

route_round_robin = 0

ipv6router = 0

ipv6forwarding = 0

ipv6_tunnel_default_mtu = 1280

ipv4_tunnel_default_mtu = 1280

(...)

subnetsarelocal = 1

tcbhashsize = 32

tcbquicklisten = 0

tcp_compat_42 = 1

tcp_cwnd_segments = 2

tcp_dont_winscale = 0

tcp_keepalive_default = 0

tcp_keepcnt = 8

tcp_keepidle = 150

```
tcp_keepinit = 150
tcp_keepintvl = 150
tcp_msl = 60
tcp_mssdflt = 536
tcpnodelack = 0
tcp_recvspace = 32768
tcp_rexmit_interval_min = 2
tcprexmtthresh = 3
tcp_rttdeflt = 3
tcp_sendspace = 32768
tcp_ttl = 60
tcptwreorder = 1
tcp_urgent_42 = 1
udpcksum = 1
udp_recvspace = 41920
udp_sendspace = 9216
udp_ttl = 30
TCP/IP>
TCP/IP>
```

Installing and Configuring TCP/IP Services

Module 3

Introduction

The focus in this module is on the installation and configuration of TCP/IP Services for OpenVMS. Some OpenVMS specifics are discussed before looking at the TCPIP\$CONFIG menu-driven utility. You do most configurations through the menus in this utility.

Objectives

Upon completion of this module, you should be able to:

- Install TCP/IP Services for OpenVMS.
- Configure TCP/IP Services for OpenVMS.

Topics

- Installation
- Configuration

Before Installation

- Register the PAK for TCP/IP for OpenVMS with one of the following commands:
 - `$@SYS$UPDATE:VMSLICENSE.COM`
 - `$LICENSE REGISTER...`
- Installation disk space required:
 - 40,000 blocks (Alpha)
 - 19,000 blocks (VAX)
- Product use disk space required:
 - 32,000 blocks (Alpha)
 - 18,000 blocks (VAX)
- During a product upgrade, existing configuration files are preserved in case you want to use them when you configure this version of the software.

Installation

Installation Sample

\$ PRODUCT INSTALL TCPIP /SOURCE=DKA400:[KITS]

The following product has been selected:

DEC AXPVMS TCPIP V5.0 Layered Product

Do you want to continue? [YES]

Configuration phase starting ...

You will be asked to choose options, if any, for each selected product and for any products that may be installed to satisfy software dependency requirements.

DEC AXPVMS TCPIP V5.0: DIGITAL TCP/IP Services for OpenVMS.

(c) Compaq Computer Corporation 1999. All rights reserved.

Compaq Computer Corporation

DIGITAL TCP/IP Services for OpenVMS offers several license options.

Do you want the defaults for all options? [YES]

Do you want to review the options? [NO] **YES**

DEC AXPVMS TCPIP V5.0: DIGITAL TCP/IP Services for OpenVMS.

Applications: YES

Are you satisfied with these options? [YES]

Execution phase starting ...

The following product will be installed:

DEC AXPVMS TCPIP V5.0 Layered Product

Portion done: 0%...10%...20%...30%...40%...50%...60%...70%...80%...90%

Portion done: 100%

The following product has been installed:

DEC AXPVMS TCPIP V5.0 Layered Product

- Do not delete any files that remain from the previous version of the product.
 - Many of these files are used by TCPIP\$CONFIG for converting your existing configuration to the new configuration.
 - Delete obsolete TCP/IP information from the Help file by running TCPIP\$CLEANUP_HELPLIB.COM.

\$ @SYS\$UPDATE:TCPIP\$CLEANUP_HELPLIB
- Reboot your system for the new TCP/IP Services software to take effect.

Displaying Information

Use the following commands to display information about installed products.

\$ product show product *

```
-----  
PRODUCT                                KIT TYPE    STATE  
-----  
DEC AXPVMS DECNET_OSI V7.2            Full LP     Installed  
DEC AXPVMS DWMOTIF V1.2-5             Full LP     Installed  
DEC AXPVMS OPENVMS V7.2               Platform    Installed  
DEC AXPVMS TCPIP V5.0-9               Full LP     Installed  
DEC AXPVMS VMS V7.2                   Oper System Installed  
-----
```

5 items found

\$

\$ product show product tcpip/full

```
-----  
PRODUCT                                KIT TYPE    STATE    MAINTENANCE  
REFERENCED BY  
-----  
DEC AXPVMS TCPIP V5.0-9    Full LP     Installed  
DEC AXPVMS OPENVMS V7.2  
-----
```

1 item found

\$

\$ product show history tcpip

```
-----  
PRODUCT                                KIT TYPE    OPERATION  DATE AND TIME  
-----  
DEC AXPVMS TCPIP V5.0-9    Full LP     Install    10-APR-1999 18:04:37  
-----
```

1 item found

\$

Configuration

- TCPIP\$CONFIG menus:
 - Make selections that enable components for your system.
 - Start or stop TCP/IP Services software.
 - Verify the configuration.

\$ @SYS\$MANAGER:TCPIP\$CONFIG
- TCPIP\$CONFIG without menus:

\$ @SYS\$MANAGER:TCPIP\$CONFIG SERVER ENABLE CLUSTER

 - Configures server components across the cluster.
 - Options include ALL, CLIENT, MINIMUM, WORKSTATION.
- Configuration files that may be created from existing files:

Existing File	New File
UCX\$SERVICE.DAT	TCPIP\$SERVICE.DAT
UCX\$HOST.DAT	TCPIP\$HOST.DAT
UCX\$NETWORK.DAT	TCPIP\$NETWORK.DAT
UCX\$ROUTE.DAT	TCPIP\$ROUTE.DAT
UCX\$PROXY.DAT	TCPIP\$PROXY.DAT
UCX\$CONFIGURATION.DAT	TCPIP\$CONFIGURATION.DAT
UCX\$EXPORT.DAT	TCPIP\$EXPORT.DAT
UCX\$PRINTCAP.DAT	TCPIP\$PRINTCAP.DAT

TCP/IP\$CONFIG

Configurable components:

- Core environment (network and transport layer software)
 - Such as interface, routing, and BIND resolver
- Client components (client application software)
 - Such as FTP, NFS, Telnet, RLOGIN, and SMTP
- Server components (server application software)
 - Such as BIND, FTP, NFS, PORTMAPPER, DHCP, NTP, and Telnet
- Other options:
 - Configure optional components (PWIP, SRI QIO, Anonymous FTP)
 - Shut down TCP/IP
 - Start up TCP/IP
 - Run IVP

\$ @sys\$manager:tcpip\$config

TCP/IP Network Configuration Procedure

This procedure helps you define the parameters required to run DIGITAL TCP/IP Services for OpenVMS on this system.
Checking TCP/IP Services for OpenVMS configuration database files.

No new database files were created.

DIGITAL TCP/IP Services for OpenVMS requires a definition for at least one interface. There are no interfaces defined on this system.

Please select the Interface option from the Core Environment Menu.

Configuration Sample

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4

Select all options.

[E] - Exit configuration procedure

Enter configuration option: **a**

DIGITAL TCP/IP Services for OpenVMS CORE ENVIRONMENT Configuration Menu

Configuration options:

- 1 - BIND Resolver
- 2 - Domain
- 3 - Routing
- 4 - Interfaces
- 5 - Time Zone

- A - Configure options 1 - 5
- [E] - Exit menu

Enter configuration option: **4**

INTERFACE Configuration

The Ethernet device(s) on your system are: EWA0: * Not Configured *
Start of configuration questions for Internet interface WE0.

WE0 is the Ethernet device EWA0:

WE0 has not been configured

* Do you want to configure WE0 [YES]:

Enter unqualified host name: den255

Enter Internet address for den255: 192.206.126.27

Enter Internet network mask for den255 [255.255.255.0]:

Enter broadcast mask for den255 [192.206.126.255]:

The following parameters will be used to define the
Internet interface WE0:

Host name: den255

Internet address: 192.206.126.27

Network mask: 255.255.255.0

Broadcast mask: 192.206.126.255

* Is the above correct [YES]:

End of configuration questions for Internet interface WE0

DIGITAL TCP/IP Services for OpenVMS CORE ENVIRONMENT Configuration Menu

Configuration options:

1 - BIND Resolver

2 - Domain

3 - Routing

4 - Interfaces

5 - Time Zone

A - Configure options 1 - 5

[E] - Exit menu

Enter configuration option: **e**

DIGITAL TCP/IP Services for OpenVMS CLIENT Components Configuration Menu

Configuration options:

1 -	FTP	Disabled
2 -	LPR/LPD	Disabled
3 -	NFS Client	Disabled
4 -	REXEC and RSH	Disabled
5 -	RLOGIN	Disabled
6 -	SMTP	Disabled
7 -	TELNET	Disabled

A - Configure options 1 - 7

[E] - Exit menu

Enter configuration option: **a**

FTP SERVER Configuration

Service is not enabled

FTP SERVER configuration options:

1 - Enable service on this node

[E] - Exit FTP_CLIENT configuration

Enter configuration option: **1**

The FTP SERVER is not enabled.

* Do you want to configure FTP SERVER [NO]: **y**

FTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

FTP SERVER configuration options:

1 - Enable service on this node

[E] - Exit FTP configuration

Enter configuration option: **1**

Creating TCPIP\$AUX identifier with a value of 3655

Creating FTP Service Entry

LPD SERVER Configuration

Service is not defined in the SYSUAF.

Nonprivileged user access is not enabled.

By default, DIGITAL TCP/IP Services for OpenVMS configures

LPD such that nonprivileged users cannot modify queue entries.

Service is not enabled

LPD SERVER configuration options:

1 - Enable service on this node

[E] - Exit LPD_CLIENT configuration

Enter configuration option: **1**

Creating SYS\$SPECIFIC:[TCPIP\$LPD]TCPIP\$PRINTCAP.DAT file.

DIGITAL TCP/IP Services for OpenVMS supports Line Printer Daemon
Protocol (see RFC 1179)

LPD requires the following:

- Name of the local queue
- Name of the remote queue
- Name of the remote host
- Spooling directory for the local queue

To add or delete printers in the TCPIP PRINTCAP database use the
\$RUN SYS\$SYSTEM:TCPIP\$LPRSETUP command

The LPD SERVER is not enabled.

* Do you want to configure LPD SERVER [NO]: **y**

LPD SERVER Configuration

Service is defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

LPD SERVER configuration options:

1 - Enable service on this node

[E] - Exit LPD configuration

Enter configuration option: 1

Creating LPD Service Entry

DIGITAL TCP/IP Services for OpenVMS supports Line Printer Daemon
Protocol (see RFC 1179)

LPD requires the following:

- Name of the local queue
- Name of the remote queue
- Name of the remote host
- Spooling directory for the local queue

To add or delete printers in the TCPIP PRINTCAP database use the

\$RUN SYS\$SYSTEM:TCPIP\$LPRSETUP command

NFS SERVER Configuration

Service is not enabled

NFS SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit NFS_CLIENT configuration

Enter configuration option: **1**

Creating TCPIP\$NOBODY Account

RSH SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

RSH SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit RSH configuration

Enter configuration option: **1**

Creating RSH Service Entry

REXEC SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

REXEC SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit REXEC configuration

Enter configuration option: **1**

Creating REXEC Service Entry

RLOGIN SERVER Configuration

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

RLOGIN SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit RLOGIN configuration

Enter configuration option: **1**

Creating RLOGIN Service Entry

SMTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Configuration is not defined in the TCPIP\$CONFIGURATION database.

Service is not enabled

SMTP SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit SMTP configuration

Enter configuration option: **1**

Creating SMTP Service Entry

TELNET SERVER Configuration

Service is not defined in the TCPIP\$SERVICE database.
Service is not enabled

TELNET SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit TELNET configuration

Enter configuration option: **1**

Creating TELNET Service Entry

DIGITAL TCP/IP Services for OpenVMS CLIENT Components Configuration Menu

Configuration options:

- 1 - FTP Enabled
- 2 - LPR/LPD Enabled
- 3 - NFS Client Enabled
- 4 - REXEC and RSH Enabled
- 5 - RLOGIN Enabled
- 6 - SMTP Enabled
- 7 - TELNET Enabled
- A - Configure options 1 - 7
- [E] - Exit menu

Enter configuration option: **e**

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Disabled
- 2 - BOOTP Disabled
- 3 - TFTP Disabled
- 4 - FTP Enabled
- 5 - LPR/LPD Enabled
- 6 - NFS Disabled
- 7 - PC-NFS Disabled
- 8 - PORTMAPPER Disabled
- 9 - TELNET/RLOGIN Enabled
- 10 - SNMP Disabled
- 11 - NTP Disabled
- 12 - METRIC Disabled
- 13 - POP Disabled
- 14 - FINGER Disabled
- 15 - RMT Disabled
- 16 - LBROKER Disabled
- 17 - DHCP Disabled
- A - Configure options 1 - 17
- [E] - Exit menu

Enter configuration option: **a**

BIND SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

BIND SERVER configuration options:

1 - Enable service on this node

[E] - Exit BIND configuration

Enter configuration option: **e**

BOOTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

BOOTP SERVER configuration options:

1 - Enable service on this node

[E] - Exit BOOTP configuration

Enter configuration option: **e**

TFTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

FTP SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit TFTP configuration

Enter configuration option: **e**

FTP SERVER Configuration

Service is defined in the SYSUAF.
Service is defined in the TCPIP\$SERVICE database.
Service is enabled on specific node.

FTP SERVER configuration options:

- 1 - Disable service on this node
- [E] - Exit FTP configuration

Enter configuration option: **e**

The FTP CLIENT is enabled.

* Do you want to configure FTP CLIENT [NO]:

LPD SERVER Configuration

Service is defined in the SYSUAF.

Service is defined in the TCPIP\$SERVICE database.

Application proxy is enabled.

By default, DIGITAL TCP/IP Services for OpenVMS configures services to require proxies for remote access. If this restriction is removed, any host on the network will have unrestricted access to LPD.

Service is enabled on specific node.

LPD SERVER configuration options:

- 1 - Disable service on this node
- 2 - Proxy access not required

[E] - Exit LPD configuration

Enter configuration option: **e**

The LPD CLIENT is enabled.

* Do you want to configure LPD CLIENT [NO]:

Approximately how many NFS Clients will be served [10]: **4**

NFS SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

NFS SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit NFS configuration

Enter configuration option: **1**

Creating NFS Service Entry

Creating MOUNT pseudo Service Entry

Creating TCPIP\$SYSTEM:TCPIP\$NFS_RUN.COM

Creating SYSS\$COMMON:[SYSEXE]TCPIP\$EXPORT.DAT

For security purposes the Export Database is created with no world access. This denies non-privileged users the use of commands which access the Export database.

* Do you want to grant world read access to non-privileged users [NO]:

The portmapper service is not configured to be enabled

WARNING: NFS will not function if the portmapper
service is disabled

You will be asked to configure the portmapper later

PCNFS SERVER Configuration

Service is not defined in the SYSUAF.
Service is not defined in the TCPIP\$SERVICE database.
Service is not enabled

PCNFS SERVER configuration options:

1 - Enable service on this node

[E] - Exit PCNFS configuration

Enter configuration option: **e**

The portmapper service is not configured to be enabled

WARNING: PC-NFS will not function if the portmapper
service is disabled

You will be asked to configure the portmapper

PORTMAPPER SERVER Configuration

Service is not defined in the SYSUAF.
Service is not defined in the TCPIP\$SERVICE database.
Service is not enabled

PORTMAPPER SERVER configuration options:

1 - Enable service on this node

[E] - Exit PORTMAPPER configuration

Enter configuration option: **1**

Creating PORTMAPPER Service Entry

Telnet and Rlogin are enabled from the client menu. If you wish to
disable the Telnet or Rlogin server then you must disable the service.

SNMP SERVER Configuration

Service is not defined in the SYSUAF.
Service is not defined in the TCPIP\$SERVICE database.
Configuration is not defined in the TCPIP\$CONFIGURATION database.

Service has been disabled: no configuration data available

SNMP SERVER configuration options:

1 - Enable service on this node

[E] - Exit SNMP configuration

Enter configuration option: 1

Creating SNMP service entry...

Creating eSNMP service entry...

Creating template file TCPIP\$EXTENSION_MIB_RUN.COM...

Creating template file TCPIP\$EXTENSION_MIB_STARTUP.COM...

Creating template file TCPIP\$EXTENSION_MIB_SHUTDOWN.COM...

This procedure allows you to configure the SNMP service from scratch, but does not allow you to modify an existing configuration.

To retain any existing SNMP configuration values enter "N" at the prompt below. If you need to check existing values exit to DCL and issue the following TCPIP command:

```
TCPIP> SHOW CONFIGURATION SNMP /FULL
```

For information on how to modify existing configuration values see the chapter on SNMP in the DIGITAL TCP/IP Services for OpenVMS Management guide.

* Do you want to enter new SNMP configuration values? [YES]:

Do you want to allow clients modify (SET) access? (Y/N) [N]: **y**
Do you want to enable authentication traps? (Y/N) [N]: **y**
Do you want to provide the public community? (Y/N) [Y]:
Do you want to provide additional community name(s)? (Y/N) [N]:
Enter contact for the system: **system**
Enter the location of the system: **Manchester, MA**

NTP SERVER Configuration

Service is not defined in the SYSUAF.
Service is not enabled

NTP SERVER configuration options:
1 - Enable service on this node
[E] - Exit NTP configuration

Enter configuration option: **1**
Creating NTP Service Entry

Creating SYS\$SPECIFIC:[TCPIP\$NTP]TCPIP\$NTP.TEMPLATE

METRIC SERVER Configuration

Service is not defined in the SYSUAF.
Service is not enabled

METRIC SERVER configuration options:
1 - Enable service on this node
[E] - Exit METRIC configuration

Enter configuration option: **1**

POP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

POP SERVER configuration options:

1 - Enable service on this node

[E] - Exit POP configuration

Enter configuration option: **1**

Creating POP Service Entry

FINGER SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

FINGER SERVER configuration options:

1 - Enable service on this node

[E] - Exit FINGER configuration

Enter configuration option: **1**

Creating FINGER Service Entry

RMT SERVER Configuration

Service is not enabled

RMT SERVER configuration options:

1 - Enable service on this node

[E] - Exit RMT configuration

Enter configuration option: **1**

LBROKER SERVER Configuration

Service is not defined in the SYSUAF.
 Service is not defined in the TCPIP\$SERVICE database.
 Service is not enabled

LBROKER SERVER configuration options:

- 1 - Enable service on this node
- [E] - Exit LBROKER configuration

Enter configuration option: **1**

Creating LBROKER Service Entry

Checking for LBROKER template file in SYS\$SYSDEVICE:[TCPIP\$LD_BKR]

Creating template SYS\$SYSDEVICE:[TCPIP\$LD_BKR]TCPIP\$LBROKER_CONF.TEMPLATE

Invalid configuration option selected, please try again

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- | | |
|-------------------|----------|
| 1 - BIND | Disabled |
| 2 - BOOTP | Disabled |
| 3 - TFTP | Disabled |
| 4 - FTP | Enabled |
| 5 - LPR/LPD | Enabled |
| 6 - NFS | Enabled |
| 7 - PC-NFS | Disabled |
| 8 - PORTMAPPER | Enabled |
| 9 - TELNET/RLOGIN | Enabled |
| 10 - SNMP | Enabled |
| 11 - NTP | Enabled |
| 12 - METRIC | Enabled |
| 13 - POP | Enabled |
| 14 - FINGER | Enabled |

- 15 - RMT Enabled
- 16 - LBROKER Enabled
- 17 - DHCP Disabled
- A - Configure options 1 - 17
- [E] - Exit menu

Enter configuration option: **e**

DIGITAL TCP/IP Services for OpenVMS Optional Components Configuration Menu

Configuration options:

- 1 - Configure PWIP Driver (for DECnet-Plus and PATHWORKS)
- 2 - Configure SRI QIO Interface (INET Driver)
- 3 - Set up Anonymous FTP Account and Directories
- A - Configure options 1 - 3
- [E] - Exit menu

Enter configuration option: e

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: e

To start up the network services:

\$

\$ @sys\$manager:tcpip\$config

TCP/IP Network Configuration Procedure

This procedure helps you define the parameters required to run DIGITAL TCP/IP Services for OpenVMS on this system.

Checking TCP/IP Services for OpenVMS configuration database files.

No new database files were created.

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **6**
Begin Startup...

```
%TCPIP-I-INFO, creating zero block size UCX compatability file:
_TCPIP-I-INFO, SYS$COMMON:[SYSEXEC]UCX$UCP.EXE
%TCPIP-I-INFO, creating zero block size UCX compatability file:
_TCPIP-I-INFO, SYS$COMMON:[SYSEXEC]UCX$SERVICE.DAT
%TCPIP-I-NORMAL, timezone information verified
%JBC-E-JOBQUEDIS, system job queue manager is not running
%JBC-E-JOBQUEDIS, system job queue manager is not running
%TCPIP$PPP-I-INFO, Loading PPP Drivers and CallBack
%RUN-S-PROC_ID, identification of created process is 000000A5
%TCPIP-I-INFO, The Internet driver and ACP were successfully loaded
%TCPIP-I-SETLOCAL, setting domain and/or local host
%TCPIP-I-STARTCOMM, starting communication
%TCPIP-I-SETPROTP, setting protocol parameters
%TCPIP-I-DEFINTE, defining interfaces
%TCPIP-I-ENABSERV, enabling services
%TCPIP-I-STARTNAME, starting name service
%TCPIP-I-LOADSERV, loading TCPIP server proxy information
```

%TCPIP-I-SERVLOADED, auxiliary server loaded with 0 proxy records
-TCPIP-I-SERVSKIP, skipped 0 communication proxy records
-TCPIP-I-SERVTOTAL, total of 0 proxy records read
%TCPIP-E-STARTERROR, error starting SMTP service
-TCPIP-E-QUEUE, error processing queue TCPIP\$SMTP_DEN255_01
-JBC-E-JOBQUEDIS, system job queue manager is not running
%TCPIP-E-STARTERROR, error starting SMTP service
-TCPIP-E-QUEUE, error processing queue TCPIP\$SMTP_DEN255_00
-JBC-E-JOBQUEDIS, system job queue manager is not running
%RUN-S-PROC_ID, identification of created process is 000000A7
%TCPIP-F-ERROR, Can not register TCP/IP
_TCPIP-F-ERROR, logical TCPIP\$INET_DOMAIN does not exist
Startup request completed.
Press Return to continue ...

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: e

\$

\$

Automatic Startup and Shutdown

- Add the following line to SYSS\$MANAGER:SYSTARTUP_VMS.COM:
@SYS\$STARTUP:TCPIP\$STARTUP
- Add the following line to SYSS\$MANAGER:SYSHUTDOWN.COM:
@SYS\$STARTUP:TCPIP\$SHUTDOWN

Introduction

This module looks more closely at the protocol stack and examines several protocol headers. Communication between the various software elements comprising the protocol stack is discussed. Several associated protocols (ARP and ICMP) are introduced as well.

Objectives

Upon completion of this module, you should be able to:

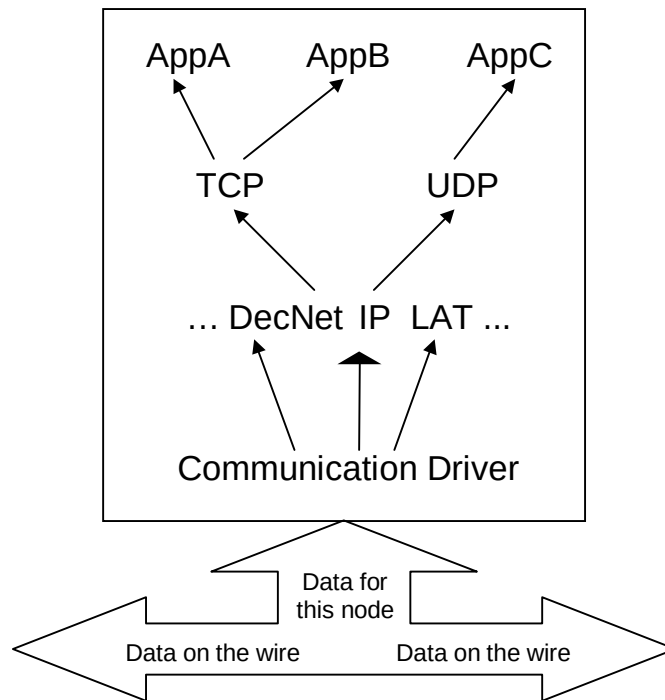
- Describe protocol headers.
- Discuss the interactions of software on the protocol stack.
- Discuss TCP, IP, UDP, ARP, and ICMP as protocols within the TCP/IP Suite.

Topics

- Data Link Protocol
- IP Header Fields
- TCP Header Fields
- ICMP and ARP
- UDP

Data Link Protocol

- The network device must issue an interrupt of the CPU when data is destined for this interface (destination MAC address in the datagram matches the address burned on the interface, or, destination MAC address is a broadcast FF-FF-FF-FF-FF-FF).
- The device should ignore data not destined for this interface.
- The driver must distribute data to appropriate network-level software modules (such as IP, LAT, and DECnet).



Network Interface Interrupts CPU and Sends Data Up the Protocol Stack

Protocol Identification

Each network-level software module must be uniquely identifiable by the device driver (bytes 20 to 21 or bytes 12 to 13 for Ethernet 802.3 and 802.2 respectively).

Protocol Types

Type	Type Byte
IP	0x0800
ARP	0x0806
DECnet Phase IV	0x6003
LAT	0x6004
SNA	0x80D5

destination addr	source addr	<u>type</u>	data	CRC
	6			6

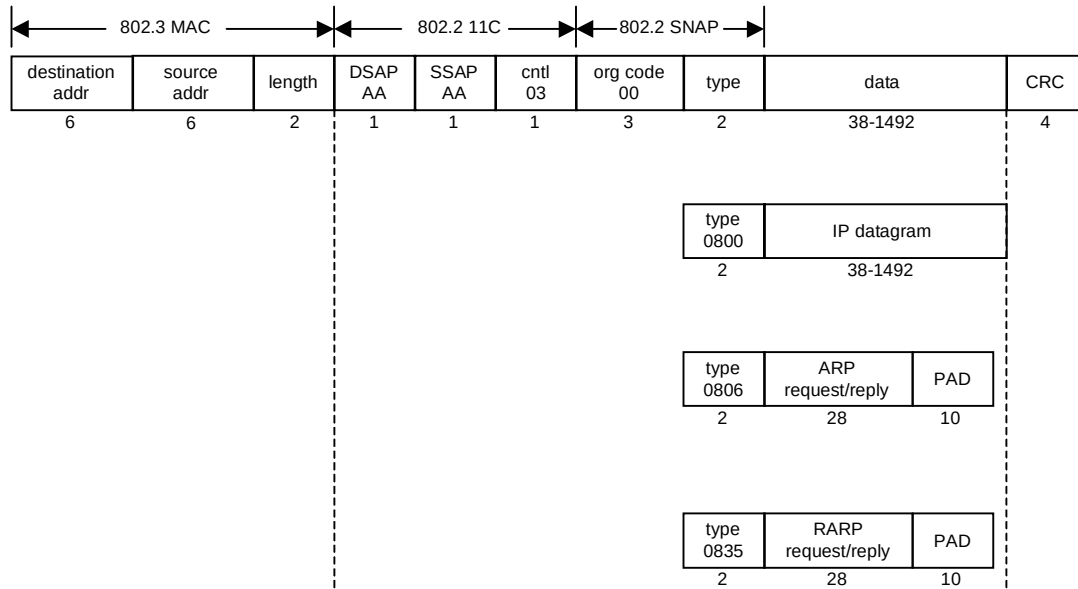
46-1500

<u>type</u>	IP datagram
0800	

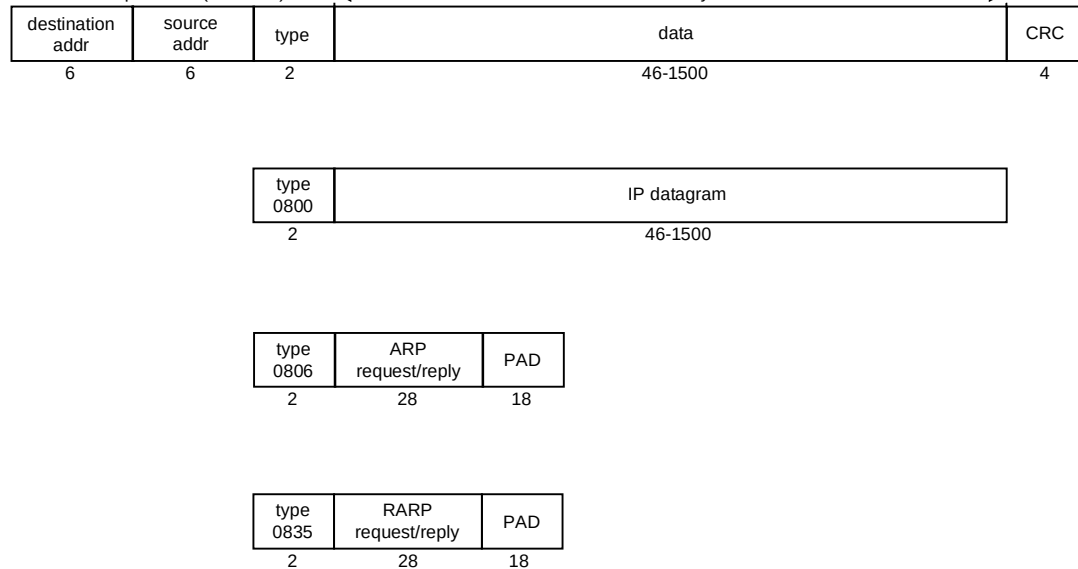
Type Byte Within Ethernet Datagram

Ethernet Encapsulation

IEEE 802.2/802.3 Encapsulation (RFC 1042):



Ethernet Encapsulation (RFC 894):

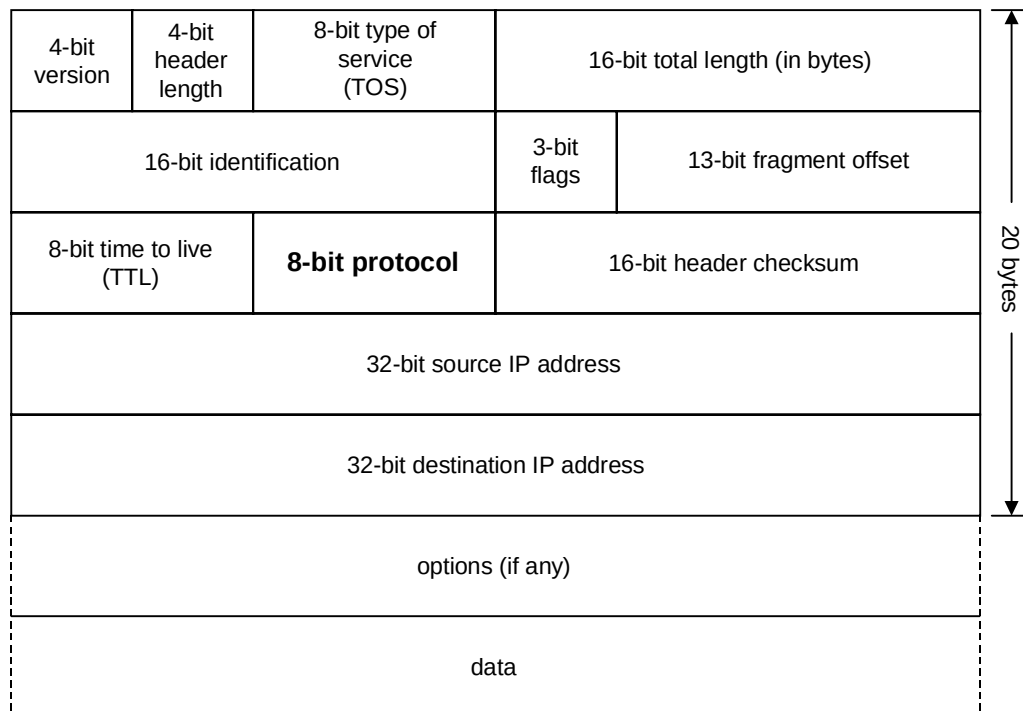


IEEE 802.2/802.3 Encapsulation (RFC 1042) and Ethernet Encapsulation (RFC 894)

IP Header

Network-level software (Internet software) must decide which transport-level software needs the data.

- Some network strategies are transport oriented (nonroutable) and skip the network level (such as LAT and NETBEUI).
- IP must decide between TCP and UDP.
- IP header contains a protocol field (octet 10) containing:
 - 6 — TCP
 - 17 — UDP

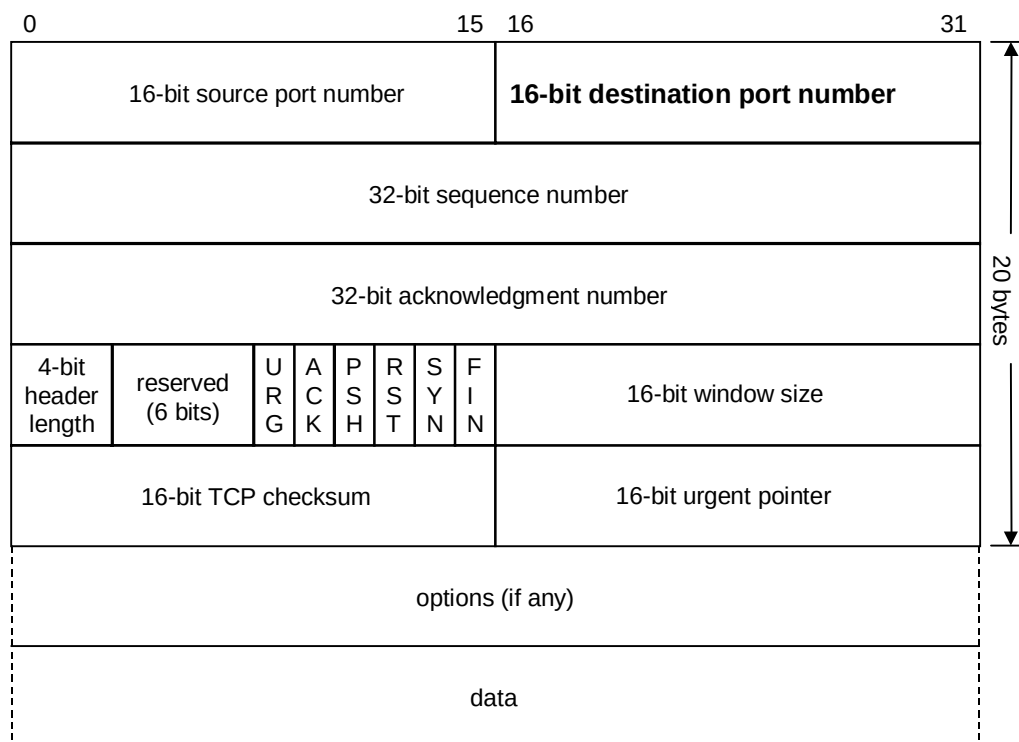


IP Datagram, Showing the Fields in the IP Header

TCP Header

Transport-level software must decide which application needs the data.

- Both TCP and UDP headers contain a 16-bit destination port number, which identifies the application (octets 2 and 3 for UDP and TCP).
 - FTP — 21
 - Telnet — 23
 - PORTMAP — 111
 - NTP — 123
 - LOGIN — 513
 - PRINTER — 515
 - ROUTE — 520
- A socket is often considered to be a combination of a 32-bit IP address and a 16-bit port address.



TCP Header

CSMA/CD

- Bus technology (such as Ethernet) has all computers sharing a common communications channel.
- Broadcast technology has all devices on the wire sense every transmission.
- Access scheme is CSMA/CD (carrier sense, multiple access/collision detect).
 - Multiple machines can potentially access the Ethernet at the same time.
 - Each machine checks for idle wire by looking for a carrier wave (carrier sense).
 - Transmissions are limited in size (approximately 1500 bytes).
 - There is a minimum idle time between transmissions.
 - Two hosts can potentially transmit inadvertently at the same time, yielding a collision (multiple access).
 - Each host delays using a binary exponential backoff policy to prevent a subsequent collision (collision detect).

Ethernet Addresses

- Ethernet is a commonly used LAN technology originally developed by Xerox, Intel, and DIGITAL.
- Ethernet is one of IEEE's multiple standards for LANs (IEEE 802.3).
- These addresses are at the lowest level recognized by hardware.
- Ethernet addresses consist of 48 bits; six fields of 8 bits each.
- Each manufacturer purchases a block of hardware addresses and applies them in sequence as they manufacture Ethernet interface hardware.

**Important**

Usually, no two hardware interfaces have the same Ethernet address!

- Unicast address is the physical address of one network interface.
- Broadcast address (all bits on) sends to all hosts.
- Multicast address sends to a subset (multicast group) of hosts.
- Interface hardware must be able to handle all three types of addresses.

The following are some examples of Ethernet addresses.

TCPIP> **arp -a**

COMPCOM (192.206.126.80) at 00-40-d0-08-a2-b3

ALFMAN (192.206.126.3) at 08-00-2b-39-80-cb

ARP command showing hardware addresses.

TCPIP>

TCPIP>

TCPIP> **ifconfig -a**

TN0: flags=80<NOARP>

WE0: flags=c43<UP, BROADCAST, RUNNING, MULTICAST, SIMPLEX>

 Hwaddr aa-00-04-00-01-04

IFCONFIG command showing hardware address.

 inet 192.206.126.27 netmask ffffffff broadcast 192.206.126.255 ipmtu 1500

LO0: flags=100c89<UP, LOOPBACK, NOARP, MULTICAST, SIMPLEX, NOCHECKSUM>

 inet 127.0.0.1 netmask ff000000 ipmtu 4096

TCPIP>

Network Standards

- Standards exist so that computers can exchange information.
- The International Standards Organization (ISO) came up with the Open Systems Interconnection (OSI) Reference Model.
- This model provides for services to be split into seven distinct layers, each with its own task.
- TCP/IP had already developed its four-layer protocol stack.
- TCP/IP has become a *de facto* standard.

Using the Protocol Stack

ISO layers as they relate to TCP/IP software:

- Application — sockets (combination of IP address and port number)
- Presentation — library routines
- Session — system calls (UNIX) and system services (OpenVMS)
- Transport – TCP (or UDP)
- Network – IP

Ethernet Frame

- Frames range from 64 bytes (octets) to 1518 bytes.
- CRC detects transmission errors — computed by the sender and recomputed and checked by the receiver.
- The type field determines which software should process the frame (network protocol), allowing multiple protocols to be used concurrently.

Bytes	Contents
0-6	Preamble consisting of 1010... for synchronization.
7	Start delimiter consisting of binary 10101011 officially starts a frame.
8-20	Two 6-byte address fields (destination, source). These are hardware addresses.
21-22	Type or length field. Used to describe type of data in the data field (Ethernet II format). Also can be used to hold the length of the data field (802.3 format).
23-(24 up to 1500)	Data field. First 3 bytes could be 802.2 header.

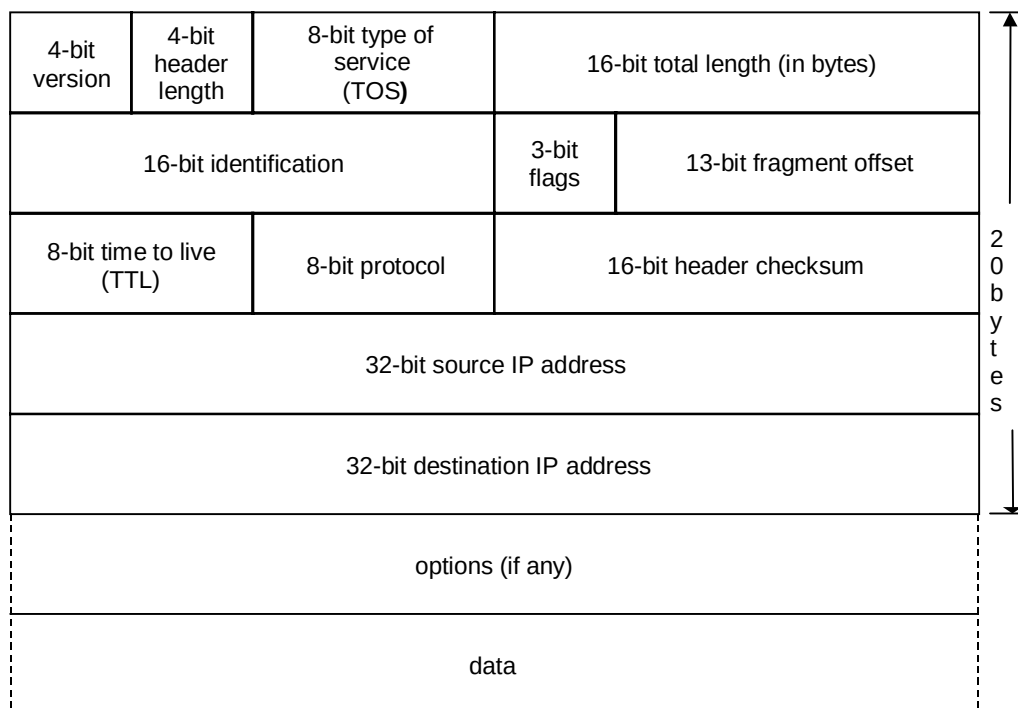
- **Pad** bytes could be used for a small packet to make sure it is at least 64 bytes.
- The above data is followed by 4 bytes of **CRC** 32 information.

IEEE 802.3 / 802.2 Frame

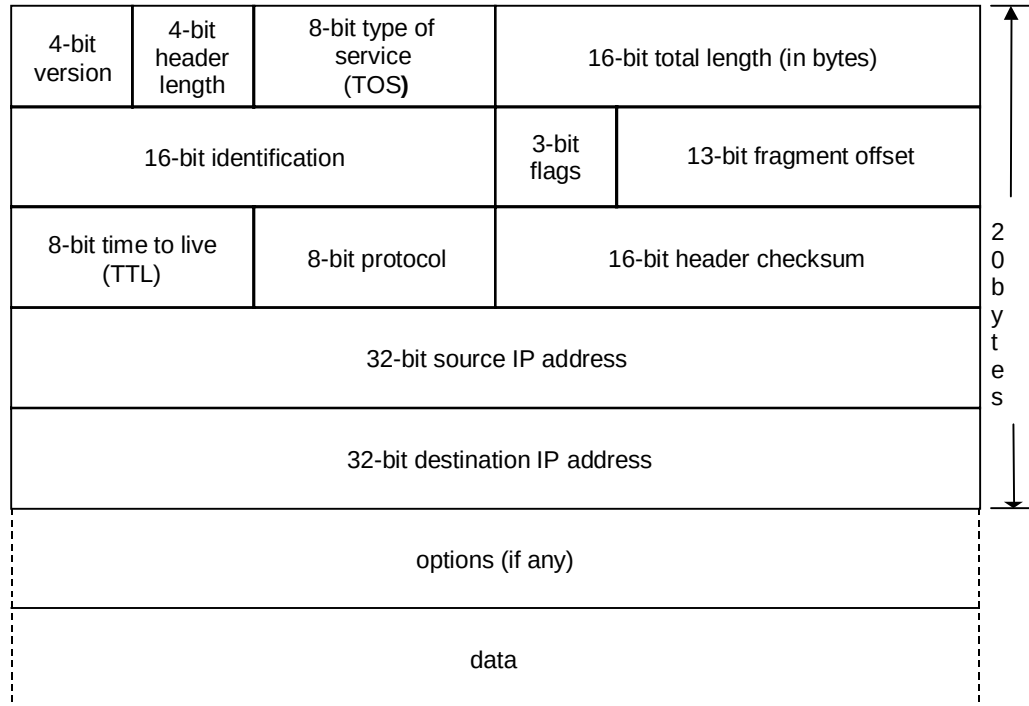
Preamble	Start Frame Delimiter	Destination Address	Source Address	Type or Length	Data Unit	Pad	FCS
----------	-----------------------	---------------------	----------------	----------------	-----------	-----	-----

Carrier Sense Multiple Access/Collision Detection (CSMA/CD)

IP Protocol Header



- TCP, UDP, ICMP, and IGMP data is transmitted as IP datagrams.
- Version is 4 (IPv4).
- **Length** is how many longwords (32-bit) are in the header.
- **Type of service (TOS)** with only 4 bits in current use.
Minimize delay, maximize throughput, maximize reliability, and minimize cost (only 1 bit can be set).
- **Total length** is the size of the datagram in bytes.
- **Identification** uniquely identifies each datagram (increments by 1).



- **Flags** indicate fragmentation position.
- **Fragmentation offset** contains the offset from the beginning of the datagram.
- **Time to live (TTL)** sets an upper limit on the number of routers through which this packet can pass.
- **Protocol** indicates to whom this should be delivered (such as TCP or UDP).
- **Header checksum** is for the header only.
- **Addresses** are the 32-bit IP addresses of source and destination.
- **Options** can be:
 - Handling restrictions for security.
 - Route recording.
 - Timestamping by routers.
 - Loose source routing (addresses through which the packet must pass).
 - Strict source routing (addresses through which the packet must solely pass).

IP Protocol TOS Field

The following table shows the recommended values for the TOS field.

Application	Minimize Delay	Maximize Throughput	Maximize Reliability	Minimize Monetary Cost	Hex Value
Telnet/Rlogin	1	0	0	0	0x10
FTP					
Control	1	0	0	0	0x10
Data	0	1	0	0	0x08
Any bulk data	0	1	0	0	0x08
TFTP	1	0	0	0	0x10
SMTP					
Command phase	1	0	0	0	0x10
Data phase	0	1	0	0	0x08
DNS					
UDP query	1	0	0	0	0x10
TCP query	0	0	0	0	0x00
Zone transfer	0	1	0	0	0x08
ICMP					
Error	0	0	0	0	0x00
Query	0	0	0	0	0x00
Any IGP	0	0	1	0	0x04
SNMP	0	0	1	0	0x04
BOOTP	0	0	0	0	0x00
NNTP	0	0	0	1	0x02



Important

Many TCP/IP implementations ignore the TOS field.

Internet Protocol Suite

- Internet protocols are a result of a Defense Advanced Research Projects Agency (DARPA) research project on network interconnection in the late 1970s.
- TCP/IP was mandated on all United States defense long-haul networks in 1983, but was not widely accepted until it was integrated with 4.2 Berkeley Software Distribution (BSD) UNIX.
- The popularity of TCP/IP is attributed to:
 - **Robust client/server framework.** TCP/IP is an excellent client/server application platform, especially in wide-area network (WAN) environments.
 - **Information sharing.** Thousands of academic, military, scientific, and commercial organizations share data, electronic mail, and services on the Internet using TCP/IP.
 - **General availability.** Implementations of TCP/IP are available on nearly every popular computer operating system. Source code is widely available for many implementations. Vendors for bridges, routers, and network analyzers all offer support for the TCP/IP protocol suite within their products.

TCP and IP Protocol

- Transmission Control Protocol and Internet Protocol:
 - Two members of the IP protocol suite.
 - IP protocol provides packet delivery for all other protocols within the TCP/IP family.
 - IP provides a best-effort, connectionless delivery system for computer data.
 - IP packets are not guaranteed to arrive at their destination, nor are they guaranteed to be received in the sequence in which they were sent.
 - The IP checksum feature confirms only the IP header's integrity.
 - Responsibility for the data contained within the IP packet (and the sequencing) is assured only by using higher-level protocols.
- The term TCP/IP is used to encompass the entire suite of protocols at all four protocol layers.

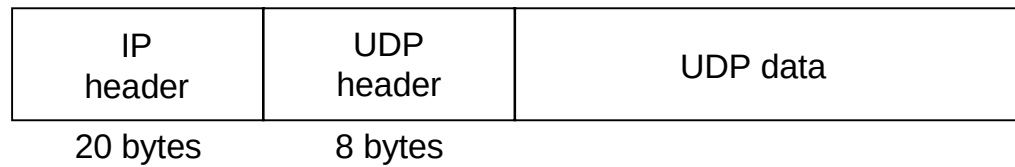
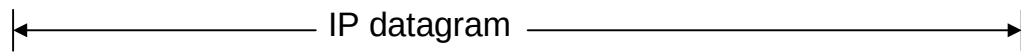
TCP Protocol

- The most common higher-level IP protocol is TCP (Transmission Control Protocol).
 - TCP supplies a reliable, connection-based protocol over (or encapsulated within) IP.
 - TCP guarantees the delivery of packets, ensures proper sequencing of the data, and provides a checksum feature that validates both the packet header and its data for accuracy.
- TCP is responsible for retransmitting a faulty packet.
 - Reliability makes TCP/IP the protocol of choice for session-based data transmission, client/server applications, and critical services such as electronic mail.
 - TCP headers require the use of additional bits to provide proper sequencing of information, as well as a mandatory checksum to ensure reliability of both the TCP header and the packet data.
 - To guarantee successful data delivery, the protocol also requires the recipient to acknowledge successful receipt of data.
 - Acknowledgments (or ACKs) generate additional network traffic, diminishing the level of data throughput in favor of reliability.

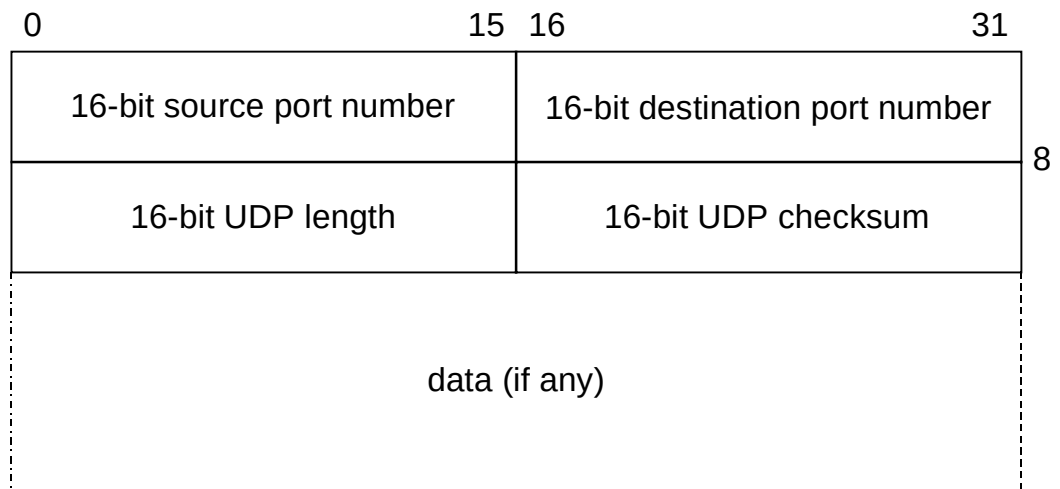
UDP

The User Datagram Protocol (UDP) provides:

- A simple transport layer protocol.
- No reliability.
- No guarantees.



UDP Encapsulation



UDP Header

- UDP, a TCP complement, offers a connectionless datagram service that guarantees neither delivery nor correct sequencing of delivered packets (much like IP).
- Higher-level protocols or applications may provide reliability mechanisms in addition to UDP/IP.
- UDP data checksums are optional, providing a way to exchange data over highly reliable networks without extra overhead.
- ACKs are also not enforced by the UDP protocol; this is left to higher-level protocols.

ARP and ICMP

- Address Resolution Protocol (ARP) and Internet Control Message Protocol (ICMP) are maintenance protocols, which are usually invisible to users and applications.
- ARP resolves an IP address to a MAC address.
 - IP packets contain both source and destination IP addresses, but the hardware address of the destination computer system must also be known.
 - IP acquires a system's hardware address by broadcasting a special inquiry packet (an ARP request packet) containing the IP address of the system with which it is attempting to communicate.
 - Nodes on the local IP network detect these broadcasts, and the system that owns the IP address in question replies by sending its hardware address to the requesting computer system in an ARP reply packet.
 - The hardware/IP address mapping is then stored in the ARP cache of the requesting system for subsequent use.
- ICMP allows two nodes on an IP network to share IP status and error information.
 - Information can be used by higher-level protocols to recover from transmission problems or by network administrators to detect network trouble.
 - Although ICMP packets are encapsulated within IP packets, they are not considered a higher-level protocol (ICMP is required in every TCP/IP implementation).
 - The ping utility makes use of the ICMP echo request and echo reply packets to determine whether a particular IP node (computer system) on a network is functional and reachable.

Introduction

TCP/IP Services for OpenVMS provides Network File System (NFS) support for transparent file system access. This module looks at the client and server sides of NFS and prepares you to configure and manage NFS.

Objectives

Upon completion of this module, you should be able to:

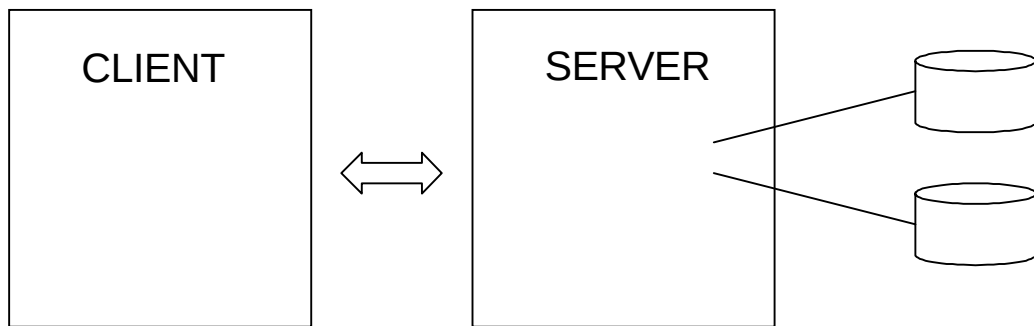
- Describe the client and server sides of NFS.
- Configure client NFS.
- Configure server NFS.
- Discuss proxy accounts.
- Discuss container files.

Topics

- NFS Concepts
- NFS Implementation
- NFS Server File Systems
- Container File System
- NFS Management Commands
- Proxy Accounts
- Client Configuration
- Client Data Manipulation
- Server Configuration
- Server Data Manipulation

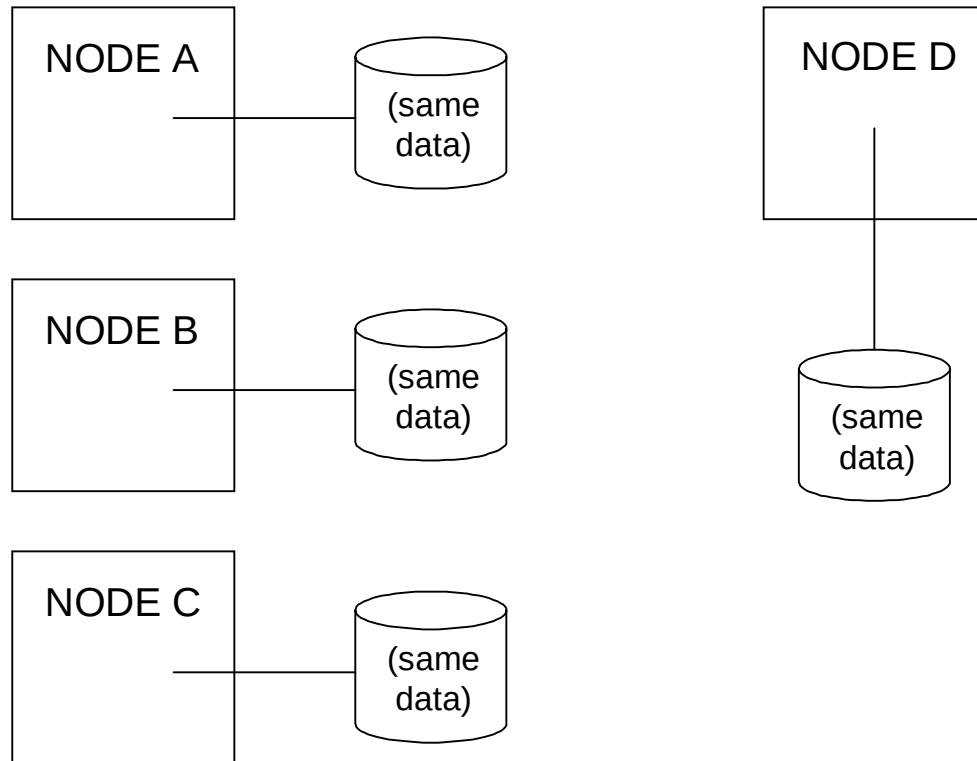
NFS Concepts

- NFS is built from two general concepts:
 - Network
 - File system
- NFS functions as a file system.
 - Applications can access the data within an NFS file system without any special case code built in.
 - Application code would remain unchanged if files were on a local file system rather than NFS.
 - NFS supports standard file system components such as directories and files.
- NFS functions over a network.
 - As a network file system, the actual files are stored on a disk local to a serving system.
 - NFS nodes that contain the exported local file systems are called NFS servers.
 - NFS nodes that access the served file systems and directories are called NFS clients.



NFS Nodes Accessing Data from Local Disks

- NFS is a file-based service allowing nodes within a network to share file system resources. NFS:
 - Shares at the file system level or directory level.
 - Prevents distribution of multiple copies of files.
 - Potentially saves on hardware (fewer disks needed).

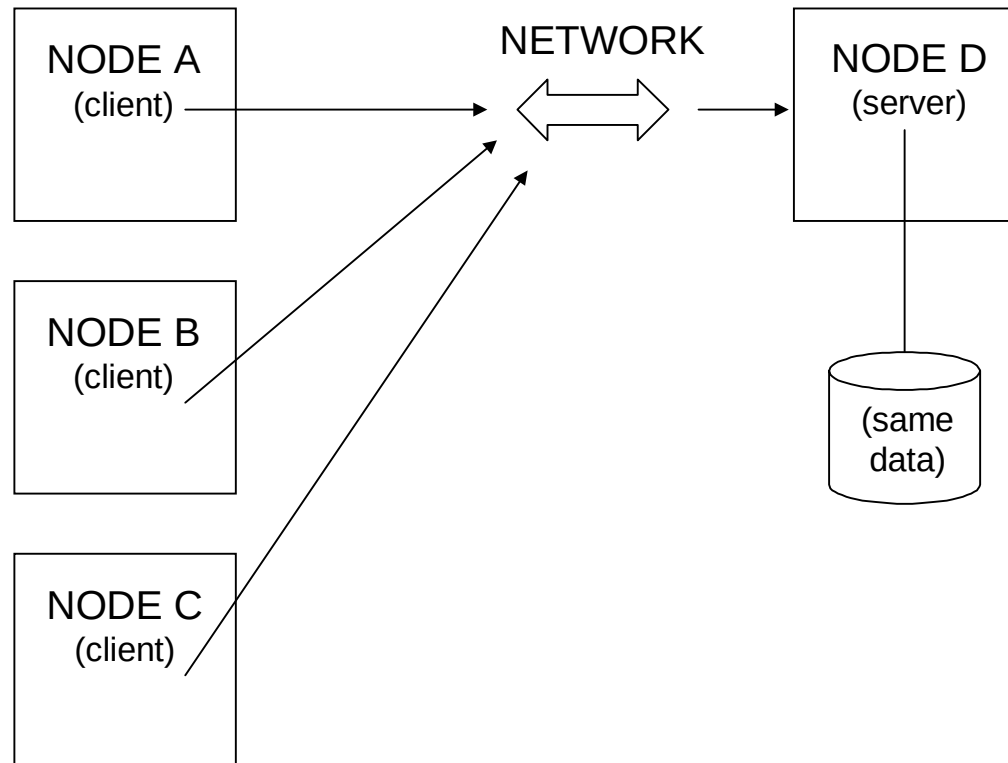


Four Nodes Duplicating Disk Storage (not Using NFS)

Client/Server Strategy

NFS is implemented through a client/server strategy.

- Server allows access to selected file systems or directories.
- Server exports file systems or directories.
- Client mounts the remote file system for access through a local directory (called a mount point).
- Users and applications on the client side have no knowledge that the file system is being served through NFS.



Three NFS Client Nodes Accessing Data over the Network

NFS Implementation

NFS was originally written by Sun Microsystems in the 1980s.

- NFS is based on RPC (remote procedure calls).
- An RPC number represents RPC functions.
- RPC numbers must be converted into port numbers.
- Perform conversion by mapping an RPC number to the corresponding port number, using PORTMAPPER software.
- Client and server run PORTMAPPER daemon process software.
- Client remotely mounts the served file system.

NFS on OpenVMS

- File compatibility issues:
 - NFS was originally designed for UNIX systems.
 - UNIX identifies users with a UID, GID pair.
 - OpenVMS identities are mapped to UNIX identities through proxies.
 - OpenVMS to OpenVMS serving involves creating proxies on both client and server; the NFS protocol demands UID and GID information.
 - OpenVMS to UNIX serving requires some preparation.
- UNIX users are mapped to an OpenVMS UIC through TCP/IP proxy accounts.
 - Can be made specifically for NFS.
 - Can be made for incoming (server) or outgoing (client) NFS activity (or both).
- A *connection* file system (not the same as a container file system) describes file system options on which NFS can work:
 - OpenVMS file systems
 - Container file systems (CFS), which mimic UNIX file systems on an OpenVMS NFS server node
- No locking supported.
- No EFS support (On-Disk Structure Level 5).
- Currently running NFS V2 (Tru64 UNIX running NFS V3).
- Uses UDP transport only (no TCP support yet).

NFS Server File Systems

- Server file systems can export two types of file systems:
 - Traditional OpenVMS file system.
 - UNIX style file system built on top of an OpenVMS file system (CFS).
- OpenVMS has a top-level directory structure called the master file directory (MFD) [000000], which contains top-level directories and reserved system files.
- UNIX has a top-level directory called the root directory (/). (A CFS also has a root directory.)

Selecting a File System

- Use an OpenVMS file system if:
 - You need to share OpenVMS file systems with another OpenVMS node.
 - You need local OpenVMS access as well as remote use.
 - You need multiple versions of files.
- Use a CFS if:
 - Your OpenVMS server is serving out files to primarily UNIX clients.
 - Clients need symbolic or hard links (UNIX specific constructs).
 - Clients expect UNIX style file names.

Container File System (CFS)

- A CFS is a UNIX style file system on an OpenVMS host that conforms to UNIX file system rules.
 - Applications that access the file system continue to work as if they were accessing files on a UNIX system.
 - An OpenVMS server can support multiple container file systems.
- A CFS provides the structure that enables users to create files.
 - All file system characteristics are stored within the container file.
 - Each file in the CFS has an associated (system-assigned) Files-11 file name.
 - An OpenVMS directory exists for each UNIX directory stored in the container.
 - The OpenVMS directory hierarchy does not duplicate the UNIX directory hierarchy.
- Because each UNIX style file is represented as an OpenVMS data file, you can use BACKUP and other standard utilities.

However, it is difficult to use DCL commands to manipulate files conveniently in a CFS.

NFS Management Commands

Server Commands

- MAP, SET CONFIGURATION MAP
- SHOW MAP, SHOW CONFIGURATION MAP
- UNMAP, SET CONFIGURATION NOMAP
- GENERATE MAP
- SET NFS_SERVER, SHOW NFS_SERVER, SHOW CFS
- CREATE EXPORT, ADD EXPORT, REMOVE EXPORT, SHOW EXPORT
- CREATE PROXY or CONVERT/VMS PROXY, ADD PROXY, REMOVE PROXY, SHOW PROXY
- CREATE DIRECTORY, DIRECTORY, REMOVE DIRECTORY, REMOVE FILE
- CREATE CONTAINER, DELETE CONTAINER
- EXPORT
- IMPORT

Client Commands

- MOUNT
- SHOW MOUNT
- DISMOUNT

Proxy Accounts

Use proxy accounts for NFS server, NFS client, PC-NFS, RSH, LPR/LPD, and user applications. Proxy accounts are held in the proxy database (TCPIP\$PROXY).

There are two types of proxies, communication proxy and NFS proxy.

- **Communication proxy** — provides an identity for users of RSH, RLOGIN, REXEC, and other utilities.
- **NFS proxy** — provides an identity for users of NFS client, NFS server, PC-NFS.
- No two proxies can have the same UID and host combination.
 - Required qualifiers: /HOST, /GID, and /UID.
 - You can combine NFS and communication proxies in one record.
 - Related commands: SHOW HOST, AUTHORIZE (DCL), CONVERT/VMS PROXY, REMOVE PROXY, and SHOW PROXY.
 - Format for NFS proxies:
ADD PROXY user { /GID=n | /HOST=host | /UID=n }
[/NFS=options]
[/PERMANENT]
- Proxy types:
 - N — NFS server (incoming)
 - O —NFS client (outgoing)
 - ON —NFS server and client
 - C — Communication
 - ND, OND — Loaded in the NFS server dynamic cache
 - CD — Loaded in the communication cache

NFS Client Configuration

\$ @sys\$manager:tcpip\$config

TCP/IP Network Configuration Procedure
(...)

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Client Components selected.

Enter configuration option: **2**

DIGITAL TCP/IP Services for OpenVMS CLIENT Components Configuration Menu

Configuration options:

- | | | | |
|---|---|---------------|-----------------|
| 1 | - | FTP | Enabled |
| 2 | - | LPR/LPD | Enabled |
| 3 | - | NFS Client | <u>Disabled</u> |
| 4 | - | REXEC and RSH | Enabled |
| 5 | - | RLOGIN | Enabled |
| 6 | - | SMTP | Enabled |
| 7 | - | TELNET | Enabled |

A - Configure options 1 - 7

[E] - Exit menu

NFS Client selected.

Enter configuration option: **3**

NFS SERVER Configuration

Software incorrectly says SERVER configuration. This configuration sequence may ask for the PORTMAPPER to be configured. NFS does not function without the PORTMAPPER running.

Service is not enabled

NFS SERVER configuration options:

1 - Enable service on this node

[E] - Exit NFS_CLIENT configuration

Enable the service.

Enter configuration option: **1**

DIGITAL TCP/IP Services for OpenVMS CLIENT Components Configuration Menu

Configuration options:

1 -	FTP	Enabled
2 -	LPR/LPD	Enabled
3 -	NFS Client	<u>Enabled</u>

All set now!

4 -	REXEC and RSH	Enabled
5 -	RLOGIN	Enabled
6 -	SMTP	Enabled
7 -	TELNET	Enabled

A - Configure options 1 - 7

[E] - Exit menu

Enter configuration option: **e**

(...)

\$

\$

\$ **tcpip**

TCPIP>

Nothing mounted.

TCPIP> **show mount**

TCPIP>

TCPIP>

TCPIP>

DNFS0: represents the next available DNFS device.

TCPIP>

TCPIP> **mount dnfs0: /transport=udp /host="alfman"**
/path="/morerz0/obrien/nfs_dir"

%TCPIP\$DNFSMOUNT-E-MOUNTFAIL, error mounting DNFS0:

-TCPIP-E-NOTCONF, NFS client not configured on this host

But I thought I configured it!!



Note

Node alfman is a Compaq Tru64 UNIX V4.0E NFS server.

TCPIP>

TCPIP> **add proxy system /nfs=outgoing /uid=0 /gid=1 /host=alfman**

Associate the system account on OpenVMS to the account with uid=0 and gid=1 on alfman (the UNIX root account). If /nfs=outgoing is eliminated from the above syntax, it would indicate both incoming (server) and outgoing (client).

TCPIP>

TCPIP> **show proxy**

VMS User_name	Type	User_ID	Group_ID	Host_name
OBRIEN	0	13	15	ALFMAN
SYSTEM	0	0	1	ALFMAN

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **5**

Shut down the network software.

Begin Shutdown...

%SYSTEM-W-NOSUCHDEV, no such device available

%JBC-E-JOBQUEDIS, system job queue manager is not running

Shutdown request completed.

Press Return to continue ...

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **6**

Restart the network software.

Begin Startup...

%TCPIP-I-NORMAL, timezone information verified

%JBC-E-JOBQUEDIS, system job queue manager is not running

%JBC-E-JOBQUEDIS, system job queue manager is not running

(...)

\$ **tcPIP**

TCPIP>

TCPIP>

TCPIP> **mount dnfs0: /host="alfman" /path="/morerz0/obrien/nfs_dir" /uid=0 /gid=1**

%TCPIP\$DNFSMOUNT-S-MOUNTED, /morerz0/obrien/nfs_dir mounted on _DNFS18:[000000]

Successful mount of a file system (directory) on alfman. The /UID=0 and /GID=1 qualifiers would not be necessary given the existence of the appropriate proxy.

Compaq recommends the use of a proxy account for remote mounts.

TCPIP>

TCPIP> **exit**

\$

\$

\$ **pipe sho sys | search sys\$input tcp**

00000114	TCPIP\$PORTM	LEF	10	110	0 00:00:00.18	305	194	N
000000A4	TCPIP\$INET_ACP	HIB	10	740	0 00:00:00.70	172	127	
000000A5	TCPIP\$NTP	LEF	9	39973	0 00:00:00.33	329	201	N
000000A6	TCPIP\$METRIC	LEF	6	18	0 00:00:06.12	130	86	
000000A9	TCPIP\$ESNMP	LEF	8	8169	0 00:00:05.15	326	204	
000000AA	TCPIP\$OS_MIBS	LEF	9	8086	0 00:00:04.92	447	140	
000000AB	TCPIP\$HR_MIB	LEF	8	14085	0 00:00:29.90	374	194	

Note PORTMAPPER.

\$

\$

\$ **pipe sho sys | search sys\$input nfs**

00000137	DNFS10ACP	HIB	10	2846	0 00:00:04.77	165	244	
----------	-----------	-----	----	------	---------------	-----	-----	--

Note NFS client daemon.

NFS Client Data Manipulation Examples

\$

\$ **sho dev d**

Device Name	Device Status	Error Count	Volume Label	Free Blocks	Trans Count	Mnt Cnt
DEN255\$DKA0:	OnLine	0				
DEN255\$DKA100:	OnLine	0				
DEN255\$DKA200:	OnLine	0				
DEN255\$DKA300:	Mounted	0	ALPHASYS	2930696	387	1
DEN255\$DKA400:	OnLine wrtlck	0				
DEN255\$DVA0:	OnLine	0				
DNFS0:	OnLine	0				
DNFS18:	Mounted	0	alfma:/morer	0	1	1

\$

Note that DNFS18: represents a remote file system on alfman. The label is shortened due to length limitations on OpenVMS disk labels.

\$

\$ **set def dnfs18:[000000]**

\$ **create OVMS_TEST1.TXT**

(...)

This file was created while logged in as system on the OpenVMS client node.

\$ **dir**

Directory DNFS18:[000000]

Files and directory are actually on the UNIX node.

DENPASSWD.;1 DISKS.;1 HOSTS.;1

OVMS_TEST1.TXT;1

Total of 4 files.

\$

\$

NFS Client File Ownership

\$

\$ **dir/prot/own**

Directory DNFS18:[000000]

Directory listing from the OpenVMS perspective.

DENPASSWD.;1	[DEFAULT]	(RWD, RWD, RD, RD)
DISKS.;1	[DEFAULT]	(RWED, RWED, RED, RED)
HOSTS.;1	[DEFAULT]	(RWD, RWD, RD, RD)
OVMS_TEST1.TXT;1	[DEFAULT]	(RWED, RWED, RED, D)

Total of 4 files.

\$

\$

\$ **rsh alfman ls -l /morerz0/obrien/nfs_dir**

Directory listing from the UNIX perspective.

-rw-r--r--	1 root	system	1104 Jun 10 15:09	denpasswd
-rwxr-xr-x	1 root	system	31114 Jun 10 15:10	disks
-rw-r--r--	1 root	system	1995 Jun 10 15:09	hosts
-rwxr-x---	1 <u>nobody</u>	system	38 Jun 11 14:26	ovms_test1.txt

Note that user *nobody* owns the file created while logged in as system.

\$ **rsh alfman tail /etc/exports**

```
# and NFS requests without UNIX authentication are considered
# anonymous; setting uid to -1 disables anonymous access.
# -access=name[:name]...
# Specifies the hosts to grant mount access to. Use of
# whitespace separated name list following options is also
# accepted. Client can be a host name, IP address, or NIS
# netgroup. If a mount access list is not provided, the path
# is exported to all hosts.
#
```

/morerz0/obrien/nfs_dir

Exported from the UNIX NFS server. NOTE: no export characteristics indicating how to handle root remote requests are specified in this file.

NFS Client Daemon I/O

\$

\$ **create nfs_test18.txt**

file lives on UNIX box.

\$

\$

\$ **sho def**

DNFS18:[000000]

\$

\$

\$ **dir/prot/own**

Directory DNFS18:[000000]

DENPASSWD.;1	[DEFAULT]	(RWD, RWD, RD, RD)
DISKS.;1	[DEFAULT]	(RWED, RWED, RED, RED)
HOSTS.;1	[DEFAULT]	(RWD, RWD, RD, RD)
<u>NFS_TEST18.TXT;1</u>	[DEFAULT]	(RWED, RWED, RED, D)
OVMS_TEST1.TXT;1	[DEFAULT]	(RWED, RWED, RED, D)

Total of 5 files.

\$

\$

\$ **type NFS_TEST18.TXT;1**

file lives on UNIX box.

\$

\$

\$ **pipe sho sys | search sys\$input nfs**

00000137	DNFS10ACP	HIB	10	<u>2934</u>	0 00:00:04.85	169	248
----------	-----------	-----	----	-------------	---------------	-----	-----

\$

\$

\$ **type NFS_TEST18.TXT;1**

file lives on UNIX box.

\$

\$

\$ **pipe sho sys | search sys\$input nfs**

00000137	DNFS10ACP	HIB	9	<u>2942</u>	0 00:00:04.86	169	248
----------	-----------	-----	---	-------------	---------------	-----	-----

Note the increase in NFS client daemon I/O counts.

NFS Client Dismount and Mount Using Proxies

\$ tcpip

TCPIP>

TCPIP> **dismount dnfs18:**

%TCPIP\$DNFSDISMOUNT-S-DISMOUNTED, DNFS18:[000000] dismounted

Dismounting an NFS mounted file system (or directory).

TCPIP>

TCPIP>

TCPIP> **add proxy system /nfs=out /uid=0 /gid=1 /host=alfman**

Associating the OpenVMS system account with uid=0 and gid=1 through a proxy.

TCPIP>

TCPIP>

TCPIP>

TCPIP> **mount dnfs0: /host="alfman" /path="/morerz0/obrien/nfs_dir" /retries=4**

%TCPIP\$DNFSMOUNT-S-MOUNTED, /morerz0/obrien/nfs_dir mounted on _DNFS19:[000000]

No need to specify /uid=0 and /gid=1 due to existence of proxy.

Without /retries=4, this mount would equate to a UNIX “hard” mount, which may cause applications to enter an RWAST state rather than reporting an error and timing out when the server becomes unavailable.

TCPIP>

TCPIP>

TCPIP> **show mount**

_DNFS19:[000000] mounted

ALFMAN:/morerz0/obrien/nfs_dir

Note that the DNFS number has increased to 19.

TCPIP>

\$

\$

UNIX Server Handling OpenVMS File Attributes

\$

\$ set def dnfs19:[000000]

\$

\$ set file /nobackup nfs_test18.txt

Alter the OpenVMS specific file characteristics to see how the UNIX server handles it.

\$

\$

\$ dir n*

Directory DNFS19:[000000]

NFSCCLIENT3.TXT;1 NFS_TEST18.TXT;1

Nothing new from the OpenVMS perspective.

Total of 2 files.

\$

\$

\$ rsh alfman ls -la /morerz0/obrien/nfs_dir

total 39

-rwxr-xr-- 1 nobody system 90 Jun 11 21:20 .\$ADFSnfs_test18.txt;1

File supporting OpenVMS characteristics.

-rw-r--r-- 1 root system 1104 Jun 10 15:09 denpasswd

-rwxr-xr-x 1 root system 31114 Jun 10 15:10 disks

-rw-r--r-- 1 root system 1995 Jun 10 15:09 hosts

-rwxr-x--- 1 nobody system 24 Jun 11 21:15 nfs_test18.txt

-rwxr-x--- 1 nobody system 0 Jun 11 21:24 nfscclient3.txt

-rwxr-x--- 1 obrien system 44 Jun 11 14:31 ovms_obrien_test1.txt

-rwxr-x--- 1 nobody system 38 Jun 11 14:26 ovms_test1.txt

Mount with Server Allowing root Access

\$

\$ **tcpip**

TCPIP> **dismount dnfs19:**

%TCPIP\$DNFSDISMOUNT-S-DISMOUNTED, DNFS19:[000000] dismounted

Dismount DNFS19.

TCPIP> **mount dnfs0: /host="alfman" /path="/morerz0/obrien/nfs_dir"**

%TCPIP\$DNFSMOUNT-S-MOUNTED, /morerz0/obrien/nfs_dir mounted on _DNFS20:[000000]

Mount on pseudo mount point DNFS0:.

TCPIP> **show mount**

_DNFS20:[000000] mounted

ALFMAN:/morerz0/obrien/nfs_dir

Now using DNFS20:.

TCPIP> **exit**

\$ **create nfs_test20.txt**

%CREATE-E-OPENOUT, error opening DNFS19:[000000]NFS_TEST20.TXT; as output

-RMS-E-CRE, ACP file create failed

-SYSTEM-F-NOTFILEDEV, device is not file structured

Default had been set to DNFS19:.

\$ **set def dnfs20:[000000]**

Change to DNFS20:.

\$ **create nfs_test20.txt**

This file should be

owned by root on the UNIX side

not the 'nobody' account.

^Z

However, that is not what happened last time!

\$

\$ rsh alfman ls -l /morerz0/obrien/nfs_dir/nfs_test20.txt

-rwxr-x--- 1 root system 78 Jun 11 21:41

/morerz0/obrien/nfs_dir/nfs_test20.txt

The file is owned by root, not *nobody*. This happened because the UNIX server allowed remote root access to take place (without reverting to the nobody account as it would by default). If the server were an OpenVMS server, proxies would arrange this.

\$ rsh alfman tail /etc/exports

(...)

/morerz0/obrien/nfs_dir -root=0

Allows root access to use UID 0, instead of the nobody account.

\$

NFS Server Configuration

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

A - Configure options 1 - 4

[E] - Exit configuration procedure

Enter configuration option: **3**

Server components selected.

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Disabled
- 2 - BOOTP Disabled
- 3 - TFTP Disabled
- 4 - FTP Enabled
- 5 - LPR/LPD Enabled
- 6 - NFS Disabled
- 7 - PC-NFS Disabled
- 8 - PORTMAPPER Enabled

(...)

A - Configure options 1 - 17

[E] - Exit menu

Enter configuration option: **6**

NFS server selected.

NFS SERVER Configuration

Service is defined in the SYSUAF.

Service is defined in the TCPIP\$SERVICE database.

Service is not enabled

NFS SERVER configuration options:

1 - Enable service on this node

[E] - Exit NFS configuration

Enter configuration option: 1

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

(...)

6	-	<u>NFS</u>	<u>Enabled</u>
7	-	PC-NFS	Disabled
8	-	PORTMAPPER	Enabled

(...)

Configuration options:

(...)

5	-	<u>Shutdown DIGITAL TCP/IP Services for OpenVMS</u>
6	-	Startup DIGITAL TCP/IP Services for OpenVMS
7	-	Run tests

A - Configure options 1 - 4

[E] - Exit configuration procedure

Enter configuration option: 5

Shut down network software.

Begin Shutdown...

(...)

Shutdown request completed.

Press Return to continue ...

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **6**

Startup network software.

Begin Startup...

%TCPIP-I-NORMAL, timezone information verified

(...)

Startup request completed.

Press Return to continue ...

NFS Server Data Manipulation Examples

Map Command

```
$
$ set def dka300:[sys0.tcpipcourse.ob_nfs]
$
$ dir
```

```
Directory DKA300:[SYS0.TCPIPCOURSE.OB_NFS]
NFS_OVMS1.TXT;1      NFS_OVMS2.TXT;1      NFS_OVMS3.TXT;1      NFS_OVMS4.TXT;1
```

Total of 4 files.

Directory to be exported.

```
$
$
$
$ type NFS_OVMS1.TXT;1
Test data in a
test file to be exported to
the UNIX world through NFS.
$
$
$
$ tcpip map "/vms_nfs" dka300:
```

Maps OpenVMS device to UNIX style name.

```
$
$ tcpip
```

TCPIP>

TCPIP> **show map**

Dynamic Filesystem Map

Pathname	Logical File System
/vms_nfs	DKA300:

TCPIP>

Exporting

TCPIP>

TCPIP>

TCPIP> **add export** "/vms_nfs/sys0/tcpipcourse/ob_nfs" /host="alfman"

Exports a directory to node alfman only. Can export to all nodes with /host=*.

An option on the ADD EXPORT command is /NAME_CONVERSION (allows the UNIX client to create files with mixed-case names and invalid OpenVMS characters).

TCPIP>

TCPIP> **show export**

File System	Host name
/vms_nfs/sys0/tcpipcourse/ob_nfs	alfman

TCPIP>

TCPIP> **exit**

\$

- User mapping
 - The NFS server can grant restricted access when the incoming UID does not proxy to an OpenVMS account.
 - The NFS server configuration creates the nonprivileged TCPIP\$NOBODY account, which can be proxied to the default user account.
 - Default user is defined by the following logical names:

\$ **show log tcpip\$Nfs*ID**

(...)

(LNM\$SYSTEM_TABLE)

"TCPIP\$NFS00000000_GID" = "-2"

"TCPIP\$NFS00000000_UID" = "-2"

\$

- Root user mapping
 - Root user (uid 0) typically is mapped to the default user.
 - To allow the remote root user to have privileged access, add a proxy record mapping uid=0 and gid=1 to a privileged OpenVMS account.

Exporting and Mapping

\$ **tcpip**

TCPIP>

TCPIP>

TCPIP>

TCPIP> **show map**

Dynamic Filesystem Map

Pathname	Logical File System
----------	---------------------

/vms_nfs	DKA300:
----------	---------

TCPIP>

TCPIP>

TCPIP> **show exports**

File System	Host name
-------------	-----------

/vms_nfs/sys0/tcpipcourse/ob_nfs	alfman
----------------------------------	--------

TCPIP>

TCPIP>

\$ **rsh alfman mount den255:/vms_nfs/sys0/tcpipcourse/ob_nfs /vms_nfs**

Client alfman mount request.

\$

\$ **rsh alfman df -t nfs**

Filesystem	512-blocks	Used	Avail.	Cap.	Mntd on
------------	------------	------	--------	------	---------

den255:/vms_nfs/sys0/tcpipcourse/ob_nfs	4110480	1179996	2930484	29%	/vms_nfs
---	---------	---------	---------	-----	----------

File Attributes

\$

\$ rsh alfman ls /vms_nfs

nfs_ovms1.txt

nfs_ovms2.txt

nfs_ovms3.txt

nfs_ovms4.txt

Directory from the UNIX client perspective showing four files created on the client.

\$

\$ set def dka300:[sys0.tcpipcourse.ob_nfs]

\$ dir

Directory DKA300:[SYS0.TCPIPCOURSE.OB_NFS]

NFS_OVMS1.TXT;1 NFS_OVMS2.TXT;1 NFS_OVMS3.TXT;1 NFS_OVMS4.TXT;1

Total of 4 files.

Directory from the OpenVMS server perspective.

\$

\$ type NFS_OVMS2.TXT;1

Test data in a
test file to be exported to
the UNIX world through NFS.

\$

\$ rsh alfman ls -l /vms_nfs

total 8

-rwxr-x---	1	nobody	nobody	78	Jun 12 15:52	nfs_ovms1.txt
-rwxr-x---	1	nobody	nobody	78	Jun 12 15:52	nfs_ovms2.txt
-rwxr-x---	1	nobody	nobody	78	Jun 12 15:53	nfs_ovms3.txt
-rwxr-x---	1	nobody	nobody	78	Jun 12 15:53	nfs_ovms4.txt

Full directory listing from the UNIX side (client).

\$

\$ **dir/prot/own**

Directory DKA300:[SYS0.TCIPPCOURSE.OB_NFS]

NFS_OVMS1.TXT;1	[1,1]	(RWED,RWED,RE,)
NFS_OVMS2.TXT;1	[1,1]	(RWED,RWED,RE,)
NFS_OVMS3.TXT;1	[1,1]	(RWED,RWED,RE,)
NFS_OVMS4.TXT;1	[1,1]	(RWED,RWED,RE,)

Total of 4 files.

Directory from the OpenVMS side (server).

\$

\$ **set prot=w:r *.***

Change permissions (protection) from the server side allowing read access to WORLD category.

\$

\$ **dir/prot/own**

Directory DKA300:[SYS0.TCIPPCOURSE.OB_NFS]

NFS_OVMS1.TXT;1	[1,1]	(RWED,RWED,RE,R)
NFS_OVMS2.TXT;1	[1,1]	(RWED,RWED,RE,R)
NFS_OVMS3.TXT;1	[1,1]	(RWED,RWED,RE,R)
NFS_OVMS4.TXT;1	[1,1]	(RWED,RWED,RE,R)

Total of 4 files.

\$

\$ **rsh alfman cat /vms_nfs/nfs_ovms2.txt**

Test data in a
test file to be exported to
the UNIX world through NFS.

\$

NFS Server Container File Manipulation Examples

\$ tcpip

TCPIP> add proxy system /uid=0 /gid=1 /host=alfman

TCPIP>

TCPIP> show proxy

VMS User_name	Type	User_ID	Group_ID	Host_name
---------------	------	---------	----------	-----------

<u>SYSTEM</u>	OND	0	1	ALFMAN
---------------	-----	---	---	--------

TCPIP>

TCPIP> create container dka300:[sys0.tcpipcourse.con_tainer] /user_name=system
/root_mode=741 /host=alfman

%TCPIP-S-CONTACREATE, container created

-TCPIP-S-DIRCREATE, directory ROOT created

-TCPIP-I-VMSUSER, OpenVMS user: SYSTEM

-TCPIP-I-UNIXID, UID: 0, GID: 1

Creating a container file system.

TCPIP> map "/test_tainer_disk" dka300:

%TCPIP-E-MAPERROR, error processing MAP or UNMAP request

-SYSTEM-F-DEVMOUNT, device is already mounted

Error creating map (already in use). You could use /vms_nfs instead of
recreating the map.

TCPIP> show map

Dynamic Filesystem Map

Pathname	Logical File System
----------	---------------------

/vms_nfs	<u>DKA300:</u>
----------	----------------

TCPIP>

TCPIP> unmap "/vms_nfs"

Unmapping a name from a device.

TCPIP> **show map**

%TCPIP-E-MAPERROR, error processing MAP or UNMAP request

-TCPIP-W-NORECORD, information not found

No more maps at this time.

TCPIP> **map "/test_tainer_disk" dka300:**

Let's make one!

TCPIP> **show map**

Dynamic Filesystem Map

Pathname

Logical File System

/test_tainer_disk

DKA300:

TCPIP>

All set now. To make the mapping permanent, you could use the set configuration map... command.

Mapping a UNIX Style Name to a Directory

TCPIP> **map** **"/test_tainer"** **dka300:[sys0.tcpipcourse.con_tainer]**

Create a map for the actual directory (as opposed to the device map).

TCPIP> **show map**

Dynamic Filesystem Map

Pathname	Logical File System
/test_tainer	DKA300:[SYS0.TCIPPCOURSE.CON_TAINER]
/test_tainer_disk	DKA300:

TCPIP>

TCPIP> **add proxy obrien /uid=13 /gid=15 /host=***

Add proxy for user name obrien.

TCPIP>

TCPIP> **show proxy**

VMS User_name	Type	User_ID	Group_ID	Host_name
OBRIEN	OND	13	15	*
SYSTEM	OND	0	1	ALFMAN

TCPIP>

TCPIP> **create directory** **"/test_tainer/obrien"** **/user_name=obrien /mode=741**
/host="alfman"

%TCPIP-S-DIRCREATE, directory obrien created

-TCPIP-I-VMSUSER, OpenVMS user: OBRIEN

-TCPIP-I-UNIXID, UID: 13, GID: 15

Create a directory within the container file system.

TCPIP>

TCPIP>

Exporting

TCPIP>

TCPIP> **show export**

File System	Host name
-------------	-----------

/vms_nfs/sys0/tcpipcourse/ob_nfs	alfman
----------------------------------	--------

TCPIP>

TCPIP>

TCPIP>

TCPIP> **show map**

Dynamic Filesystem Map

Pathname	Logical File System
----------	---------------------

/test_tainer	DKA300:[SYS0.TCPIPCOURSE.CON_TAINER]
--------------	--------------------------------------

/test_tainer_disk	DKA300:
-------------------	---------

TCPIP>

TCPIP>

TCPIP> **add export "/test_tainer" /host=***

Export the container file system.

TCPIP>

TCPIP> **show export**

File System	Host name
-------------	-----------

/test_tainer	*
--------------	---

/vms_nfs/sys0/tcpipcourse/ob_nfs	alfman
----------------------------------	--------

TCPIP>

TCPIP>

TCPIP>

TCPIP> **exit**

\$ rsh alfman mount den255:/test_tainer /test_tainer

Mount from the client side.

\$

\$ **rsh alfman df -t nfs**

Filesystem	512-blocks	Used	Available	Capacity	Mounted on
den255:/test_tainer	4110480	1180052	2930428	29%	/test_tainer

\$

Introduction

TCP/IP Services for OpenVMS provides SLIP and PPP support for connecting through serial lines. This module introduces both protocols and presents the commands used to configure the serial lines.

Objectives

Upon completion of this module, you should be able to:

- Describe serial line options.
- Distinguish PPP from SLIP.
- Set up a serial line using SLIP.

Topics

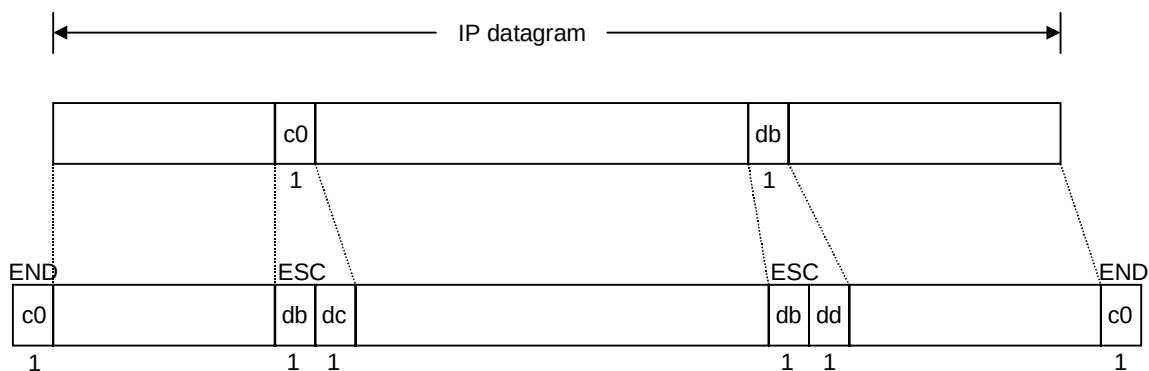
- Serial Connections
- SLIP Encapsulation
- PPP Encapsulation
- Setting Up SLIP

Serial Connections

- You can connect two systems using serial lines.
 - Implemented through a point-to-point connection.
 - Typically involves a serial cable and a terminal hardware port on the two systems.
 - May involve modems and telephone lines.
- The network software must be aware of the intent to use the terminal port for a network connection.
 - Upper layer protocols remain the same.
 - Network access layer is through the terminal driver.
- Data is arranged using one of two simple protocols:
 - SLIP (actually CSLIP)
 - PPP

SLIP Encapsulation

- SLIP encapsulation is a simple form of encapsulation for IP datagrams on serial lines.
 - Bit sequence hex c0 is sent to mark the end of a datagram (and the beginning of the next one).
 - Hex db dc is inserted if the real data matches the end character c0.
 - Hex db dd is inserted if the real data matches the db sequence.
- CSLIP
 - To carry 1 byte of data (perhaps due to Telnet or rlogin) would require a 20-byte IP header and a 20-byte TCP header.
 - CSLIP reduces that overhead to 3 to 5 bytes.
 - CSLIP maintains the state of the connection and assumes that the header information will not change much.



PPP Encapsulation

The Point-to-Point Protocol (PPP):

- Corrects deficiencies in SLIP.
- Adds a CRC field.
- Supports more than just IP.
- Uses TCP and IP header compression (like CSLIP).
- Provides 3 bytes extra overhead per frame.
- Should eventually replace SLIP.
- Uses a protocol field to identify the network layer protocol (DECnet, IP, or OSI).
- Begins and ends with hex 7e.

flag 7E	addr FF	control 03	protocol	information	CRC	flag 7E
1	1	1	2	up to 1500 bytes	2	1

protocol 0021	IP datagram
------------------	-------------

protocol C021	link control data
------------------	-------------------

protocol 8021	network control data
------------------	----------------------

Format of PPP Frames

PPP Configuration

- Configure PPP with one of the following commands.
 - Valid until next net restart:
\$ TCPIP SET INTERFACE pp0
 - Permanent:
\$ TCPIP SET CONFIGURATION INTERFACE pp0
/COMPRESS
/DESTINATION
/HOST
/NETWORK_MASK
/SERIAL_DEVICE
- PPP can function as a:
 - Dialup provider or client.
 - Straight serial connection.
- Access modem-specific commands through the DCL command PPPD.

Setting Up SLIP

**Note**

This sequence assumes that the other end of the serial line is a Tru64 UNIX system already configured to use SLIP through the appropriate UNIX commands (such as `netsetup`, `netconfig`, `sysman`, and `startslip`).

\$

\$ **tcpip**

TCPIP>

TCPIP> **show inter sl0**

%TCPIP-W-NODEVINTE, interface not found

No sl0 interface is configured yet.

TCPIP>

TCPIP> **ping 192.206.127.3**

PING 192.206.127.3 (192.206.127.3): 56 data bytes

Note the new network number (192.206.127).

ping: sendto:

no route to host

----192.206.127.3 PING Statistics----

3 packets transmitted,

0 packets received,

100% packet loss

Cannot get there, not configured.

TCPIP>

TCPIP> **set inter sl0 /host=192.206.127.27 /network_mask=255.255.255.0
/serial_device=tta0:**

Create interface sl0 and associate it with TTA0:.

TCPIP>

TCPIP> **sho inter sl0**

Interface	IP_Addr	Network mask	Packets		MTU
			Receive	Send	
SL0	192.206.127.27	255.255.255.0	0	0	1006

TCPIP>

TCPIP> **ping 192.206.127.3**

PING 192.206.127.3 (192.206.127.3): 56 data bytes

ping: sendto:

no route to host

----192.206.127.3 PING Statistics----

3 packets transmitted,

0 packets received,

100% packet loss

Still cannot get there.

TCPIP> **show route**

No route to network 192.206.127.

DYNAMIC

Type	Destination	Gateway
DH	127.0.0.1	127.0.0.1
AN	192.206.125.0/24	192.206.126.3
DN	192.206.126.0/24	192.206.126.27
AN	192.206.126.0/24	192.206.126.3

TCPIP>

TCPIP>

TCPIP>

TCPIP> **netstat -rn**

UNIX command showing route table info.

Routing tables

Destination	Gateway	Flags	Refs	Use	Interface
-------------	---------	-------	------	-----	-----------

Route Tree for Protocol Family 2

127.0.0.1	127.0.0.1	UH	3	1500	LO0
192.206.125/24	192.206.126.3	UGS	0	0	WE0
192.206.126/24	192.206.126.27	U	3	1494	WE0
192.206.126/24	192.206.126.3	UGS	0	0	WE0

TCPIP>

TCPIP> **set route /default /gateway=192.206.127.27**

Add a default route.

TCPIP>

TCPIP>

TCPIP> **route add default 192.206.127.27**

UNIX style command to add a default route.

(...)

TCPIP>

TCPIP>

TCPIP> **show route**

DYNAMIC

Type	Destination	Gateway
------	-------------	---------

DN	<u>0.0.0.0</u>	192.206.127.27
----	----------------	----------------

Default route is represented as 0.0.0.0.

```

AH    127.0.0.1                127.0.0.1
DN    192.206.125.0/24         192.206.126.3
AN    192.206.126.0/24         192.206.126.27
DN    192.206.126.0/24         192.206.126.3

```

TCPIP>

TCPIP> **netstat -rn**

UNIX style display of the routing table.

Routing tables

Destination	Gateway	Flags	Refs	Use	Interface
-------------	---------	-------	------	-----	-----------

Route Tree for Protocol Family 2

<u>default</u>	192.206.127.27	US	0	0	SL0
127.0.0.1	127.0.0.1	UH	3	1508	LO0
192.206.125/24	192.206.126.3	UGS	0	0	WE0
192.206.126/24	192.206.126.27	U	3	1502	WE0
192.206.126/24	192.206.126.3	UGS	0	0	WE0

Represents default route with default.

TCPIP>

TCPIP> **ping 192.206.127.3**

PING 192.206.127.3 (192.206.127.3): 56 data bytes

64 bytes from 192.206.127.3: icmp_seq=0 ttl=64 time=184 ms

64 bytes from 192.206.127.3: icmp_seq=1 ttl=64 time=183 ms

64 bytes from 192.206.127.3: icmp_seq=2 ttl=64 time=183 ms

64 bytes from 192.206.127.3: icmp_seq=3 ttl=64 time=183 ms

----192.206.127.3 PING Statistics----

4 packets transmitted,
4 packets received,
0% packet loss
round-trip (ms) min/avg/max = 183/183/184 ms

All set now. Note the slow time statistic.

TCPIP>

TCPIP> **sysconfig -q inet ipforwarding**

inet:

ipforwarding = 0

Using sysconfig to see if the system is configured as a router (0 means no).

TCPIP>

TCPIP>

TCPIP> **set protocol ip /forward**

Ready to forward packets.

TCPIP>

TCPIP>

TCPIP> **sysconfig -q inet ipforwarding**

inet:

ipforwarding = 1

TCPIP>

TCPIP>

TCPIP> **netstat -rn**

Routing tables

Destination	Gateway	Flags	Refs	<u>Use</u>	Interface
-------------	---------	-------	------	------------	-----------

Route Tree for Protocol Family 2

default	192.206.127.27	US	0	<u>4</u>	SL0
127.0.0.1	127.0.0.1	UH	3	1518	LO0
192.206.125/24	192.206.126.3	UGS	0	0	WE0
192.206.126/24	192.206.126.27	U	3	1512	WE0
192.206.126/24	192.206.126.3	UGS	0	0	WE0

Route Tree for Protocol Family 26

Check use count statistic for SL0.

TCP/IP>

TCP/IP> **exit**

\$

\$

\$ **rsh alfman "/sbin/ifconfig sl0"**

sl0: flags=11<UP,POINTOPOINT>

inet 192.206.127.3 --> 192.206.127.27 netmask fffffff00 ipmtu 296

View from the Tru64 UNIX side.

\$

\$

\$ **rsh alfman "/sbin/ping 192.206.127.27"**

PING 192.206.127.27 (192.206.127.27): 56 data bytes

64 bytes from 192.206.127.27: icmp_seq=0 ttl=64 time=185 ms

64 bytes from 192.206.127.27: icmp_seq=1 ttl=64 time=183 ms

64 bytes from 192.206.127.27: icmp_seq=2 ttl=64 time=183 ms

%RSH-S-LCLCLOSED, Local connection closed

Ping from the UNIX side.

\$

\$

\$ **tcpip "netstat -rn"**

Routing tables

Destination	Gateway	Flags	Refs	Use	Interface
-------------	---------	-------	------	-----	-----------

Route Tree for Protocol Family 2

default	192.206.127.27	US	0	<u>14</u>	SL0
127.0.0.1	127.0.0.1	UH	3	1540	LO0
192.206.125/24	192.206.126.3	UGS	0	0	WE0
192.206.126/24	192.206.126.27	U	3	1609	WE0
192.206.126/24	192.206.126.3	UGS	0	0	WE0

Note SL0 use count after successful ping.

\$

\$ **show dev tta0:/full**

Terminal TTA0:, device type unknown, is online, record-oriented device, carriage control, network device.

Error count	0	Operations completed	8
Owner process	"TCP/IP\$INET_ACP"	Owner UIC	[SYSTEM]
Owner process ID	000000A4	Dev Prot	S:RWPL,0:RWPL,G,W
Reference count	1	Default buffer size	80

TTA0: is being used as the serial line port.

\$

\$ **tcpip**

TCP/IP>

TCP/IP> **set nointer sl0**

Remove the sl0 interface.

TCP/IP>

If using a modem, be sure to set the terminal characteristics as follows:

\$ **set terminal tta0: /altypeahd /autobaud /dialup /disconnect /eightbit /modem /nohangup /nohostsync /nopasthru /noreadsync /nottysync /permanent /typeAhead**

To make sure you can log in to the system:

\$ **set host /dte tta0:**

If you have multiple failures while testing, be prepared to clear the security information in your intrusion record (you need SECURITY privilege).

\$ **show intrusion**

\$ **delete/intrusion_record**

Introduction

This module traces the activities involving routing. TCP/IP Services for OpenVMS provides both static and dynamic routing. This module prepares you for more complex routing scenarios by looking carefully at some simple cases. The ROUTED daemon process is discussed as a component of dynamic routing.

Objectives

Upon completion of this module, you should be able to:

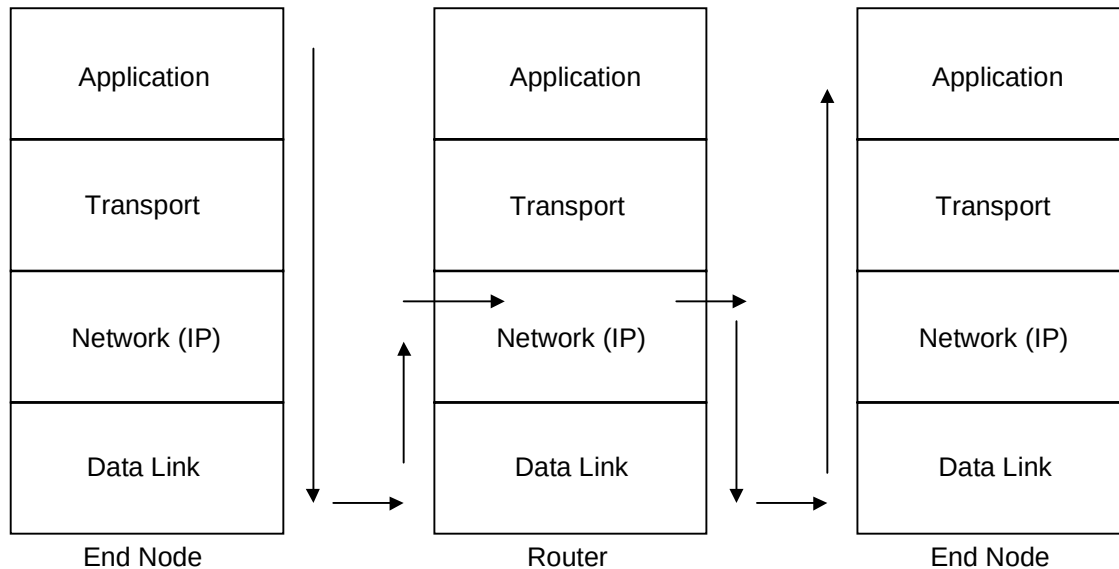
- Describe routing activities.
- Distinguish between static and dynamic routing.
- Manage the routing table.
- Configure dynamic routing.

Topics

- Routing Introduction
- Routing Table
- Static Routing
- Dynamic Routing
- ROUTED
- GATED
- Configuring Dynamic Routing

Routing

- Routing is the process of getting a packet of data from node A to node B.
- Routing uses IP based software on each router node.



Router Node Bringing Datagram Up to Network Level Only

- Routers receive an IP datagram and forward it to its ultimate destination based on the destination address in the IP header.
- Datagrams arrive at the router based on MAC level addressing.
- Destination address in the IP header does not change as the datagram visits routers.
- MAC address changes after IP header's destination address is found to be other than the router node's IP address.
- IP software examines the routing table and decides where to send the datagram next.

Router or Gateway

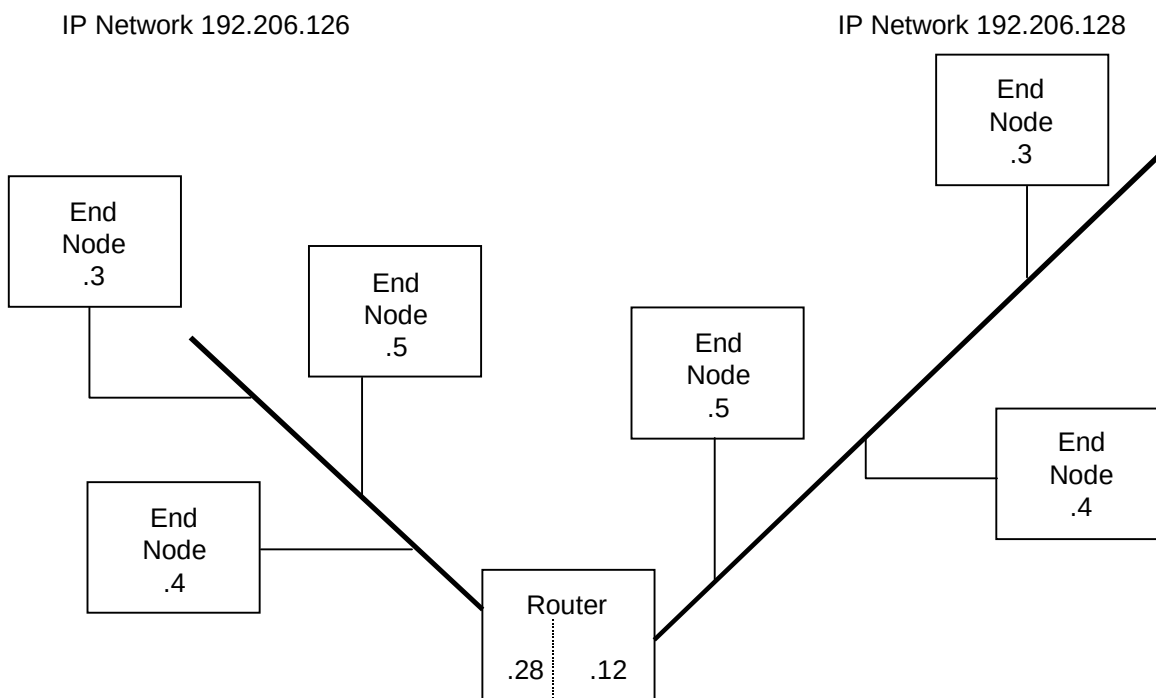
A network consists of end nodes (nonrouters) and routers.

- TCP/IP documentation uses the terms *router* and *gateway* interchangeably.
- ISO describes a gateway as a component that alters the header information surrounding the data in preparation for it being placed on a different (non-TCP/IP) style of network.

Note

Routing is **not** accomplished through ROUTED or GATED processes.

- These dynamic routing processes are responsible for keeping useful information in the routing table.

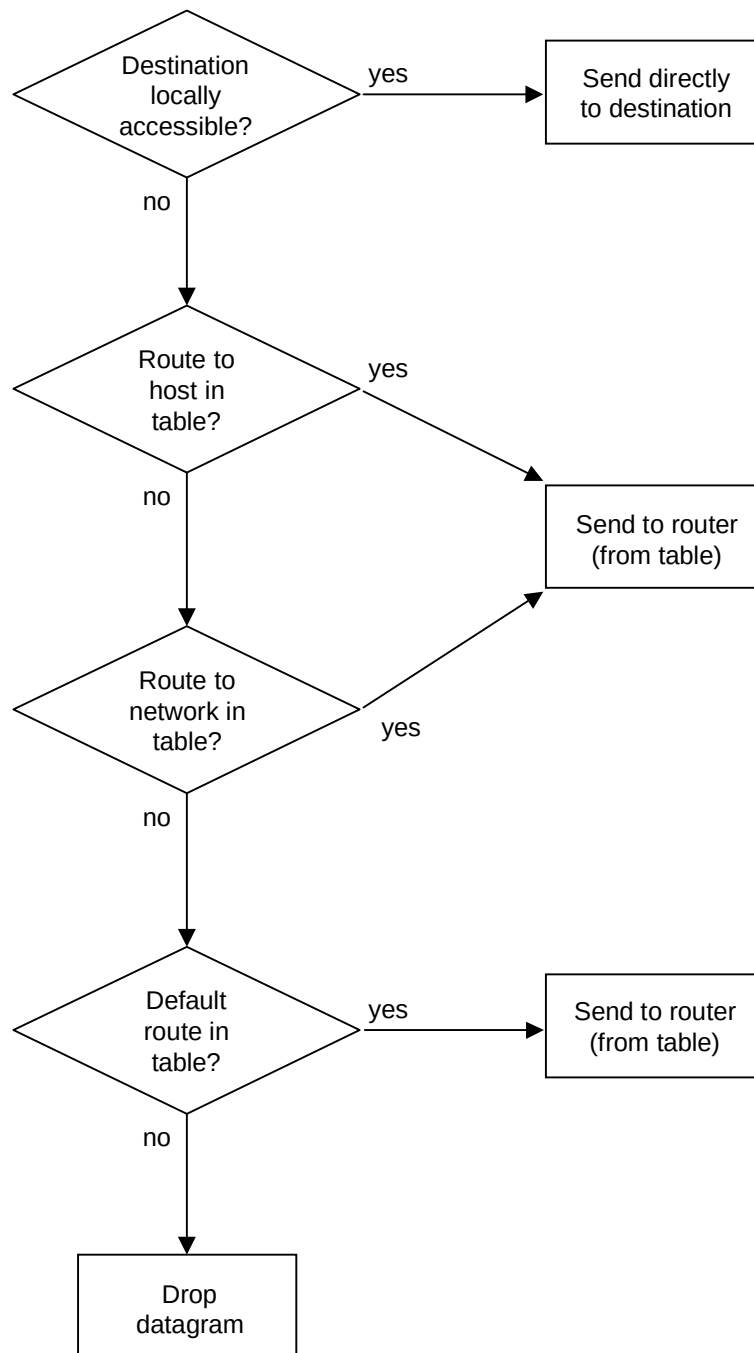


Router with Interfaces on Two Networks

Routing Table Decisions

- An application wants to send some data to port 4032 on node 192.206.126.33. The address of the sending node is 16.23.34.45.
- IP software on the sending node checks to see if the destination address is accessible from a local interface (a local network).
 - If so, the datagram is placed on the wire.
 - If not, the routing table is checked to see if there is an entry describing where to send the datagram to route it to the destination for **this particular (destination) node**.
 - If nothing describes how to handle a datagram for this destination node, check the routing table to see if there is an entry describing what to do for **this particular (destination) network (or subnet)**.
 - If nothing describes what to do for this particular network (or subnet), check the routing table for a **default entry**.
 - If none of the above options work, report an error and drop the datagram.

Router Decision Process



Routing Table

- The routing table is an in-memory structure holding information that describes how to send a datagram in the direction of its destination node.
- Routing types:
 - **Static routing** — Routing table entries are usually made by hand and are only changeable by hand.
 - **Dynamic routing** — Routing table entries are created and deleted automatically based on information retrieved from ROUTED or GATED daemon processes on adjacent routers (sometimes called *adaptive routing*).
- Routing can be seeded at network startup from information in a file:
 - TCPIP\$ROUTE for OpenVMS
 - /etc/routes for UNIX
- Altering data in the routing table solves most routing problems.
- All nodes—end nodes and router nodes—have a routing table.
 - Router nodes have a larger and more complex routing table.
 - To view a routing table, enter one of the following commands:
 - ♦ \$ TCPIP show route
 - ♦ \$ TCPIP "netstat -rn"

TCPIP> **show route**

DYNAMIC

Type	Destination	Gateway
------	-------------	---------

DH	127.0.0.1	127.0.0.1
----	-----------	-----------

AN	192.206.128.0/24	192.206.128.27
----	------------------	----------------

Routing table display using an OpenVMS style command.

TCPIP>

TCPIP> **netstat -rn**

Routing tables

Destination	Gateway	Flags	Refs	Use	Interface
-------------	---------	-------	------	-----	-----------

Route Tree for Protocol Family 2

127.0.0.1	127.0.0.1	UH	3	184	L00
-----------	-----------	----	---	-----	-----

192.206.128/24	192.206.128.27	U	1	176	WE0
----------------	----------------	---	---	-----	-----

Routing table display using a UNIX style command.

TCPIP>

Static Routing

- When the network has few routers and is stable, nondynamic routing is useful because:
 - It introduces little overhead and ties up little bandwidth.
 - It is less flexible and less convenient than dynamic routing.
- SET ROUTE (or route add) creates a static route.
 - If TCPIP is not active, SET ROUTE modifies the permanent database.
 - If TCPIP is active, SET ROUTE modifies the volatile database.
 - If TCPIP is active, SET ROUTE/PERMANENT updates the permanent database.
- To define a route to **any** host on a network, enter the following commands.
 - OpenVMS:

```
TCPIP> SET ROUTE network_address /GATEWAY="router_address"
      /NETWORK /MASK=non_default_subnet_mask
```
 - UNIX style:

```
TCPIP> route add -net network_address router_address
      -netmask non_default_subnet_mask
```
- To define a route to a **specific** host on a network, enter the following commands.
 - OpenVMS:

```
TCPIP> SET ROUTE host_address /GATEWAY="gateway"
```
 - UNIX style:

```
TCPIP> route add -host host_address router_address
      -netmask non_default_subnet_mask
```

TCPIP\$ROUTE

- At startup, the information in the permanent routing database file (if any exists) is loaded into the volatile routing database.
 - You can add permanent routes while the network is either stopped or running.
 - While running, use the /PERMANENT qualifier.
- The database is pointed to by TCPIP\$ROUTE.
- A **default route** is used when no other information in the table provides the necessary direction.

To define a default route, use the /DEFAULT qualifier:

- OpenVMS:
TCPIP> SET ROUTE /**DEFAULT** /GATEWAY=router_address
- UNIX style:
TCPIP> route add **default** router_address

Displaying Static Routes

To display static routes, use the `SHOW ROUTE` command (or `netstat -rn`).

- To see only the permanent database, specify the `/PERMANENT` qualifier.
- The display shows the following types of routes:
 - N — network route
 - H — host route
 - A — automatically created route (after entering `SET INTERFACE`)
 - P — permanent
 - D — dynamically redirected by ICMP (not by `ROUTED` or `GATED`)

TCPIP> **show route**

DYNAMIC

Type	Destination	Gateway
DN	0.0.0.0	192.206.127.27
AH	127.0.0.1	127.0.0.1
DN	192.206.128.0/24	192.206.128.27

TCPIP>

Dynamic Routing

- Complex environments require a more flexible approach to routing than a static routing table provides.
- Routing processes using dynamic routing protocols distribute information that reflects changing network conditions and updates the routing table accordingly. With dynamic routing, you can:
 - Switch to an alternate route when a primary route becomes unavailable.
 - Determine the best route to a given destination.
- Routing daemon processes implement a routing protocol, that is, the set of rules that decide which routes go into the routing table.
- The kernel generates messages when events occur that may be of interest to the dynamic routing daemon process (interface down).
- Compaq TCP/IP Services for OpenVMS implements two routing daemons:
 - Routing Daemon (ROUTED)
 - Gateway Routing Daemon (GATED)

Routing Daemon (ROUTED)

- The routing daemon supports the Routing Information Protocol (RIP).
 - ROUTED daemons send broadcast messages out each interface on the node on which they are running.
 - The daemon allows adjacent ROUTED daemons to be informed about the network information it has in its routing table.
 - The daemon repeats the messages every 30 seconds.
- A ROUTED daemon adds a route to its routing table when the incoming advertised route is better than the one that is currently in the table.
- A ROUTED daemon removes entries from its table if it does not hear a repeat confirmation within three minutes.
- Decisions are based on hop count.
 - Hop count is the count of routers between two nodes.
 - Hop count of 16 = infinity.
- ROUTED issues routing update requests then listens for responses.
 - A system configured to supply RIP information responds to the request with an update packet.
 - The update packet contains destination addresses and routing metrics (hop counts) associated with each destination.
 - After receiving a RIP update, the ROUTED might use the information to update its routing table.

Gateway Routing Daemon (GATED)

GATED is a newer dynamic routing daemon that:

- Supports more than just the RIP protocol (such as OSPF, EGP, and BGP).
- Supports interior and exterior gateway protocols.
- Has an associated configuration file.

GATED Supported Protocols

- **Routing Information Protocol (RIP) 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. RIP is a distance-vector protocol.
- **Open Shortest Path First (OSPF) Version 2**

Another interior routing protocol, OSPF is a link-state protocol (shortest path first) and is better suited than RIP for use in complex networks with many routers.
- **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.
- **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.
- These routing protocols are configured in the GATED configuration file TCPIP\$GATED.CONF. This file contains statements that control tracing options, select routing protocols, and manage routing information.

Enabling and Disabling Dynamic Routing

TCPIP\$CONFIG enables dynamic routing and configures your host to receive routing protocol messages.

- To enable dynamic routing:
 1. Choose the *Routing* option from the CORE ENVIRONMENT menu.
 2. “Do you want to reconfigure dynamic ROUTED or GATED routing [NO]:” **YES**
 3. If enabling ROUTED, your node can supply RIP updates to other hosts on the network (in addition to receiving RIP updates).
 4. If enabling GATED, configure the routing protocols in the GATED configuration file TCPIP\$GATED.CONF.
- To disable dynamic routing:
 1. Choose *Routing* from the CORE ENVIRONMENT menu.
 2. “Do you want to disable GATED routing configuration [NO]:” **YES**
or enter:
TCPIP> **STOP ROUTING**

Configuring GATED

- Configure GATED before enabling GATED routing.
 - Copy and edit the sample file TCPIP\$GATED.TEMPLATE (located in SYS\$SYSDEVICE:[TCPIP\$GATED]).
 - Add statements that select routing protocols and manage routing information.
 - Save it to TCPIP\$GATED.CONF, in SYS\$SYSDEVICE:[TCPIP\$GATED].
- Use TCPIP\$CONFIG to enable GATED.
 - If GATED is already running, you must stop and restart it.
\$ TCPIP STOP ROUTING/GATED
\$ TCPIP START ROUTING/GATED
 - GATED terminates if there are errors in the file.

Sample GATED Configuration File

```
#-----
#
#       Copyright (c) Digital Equipment Corporation, 1998
#
#       TCPIP$GATED.CONF - Sample gateway routing daemon (GATED)
#       configuration file, preconfigured for RIP v1.
#
# This file contains information that is read by the GATED daemon at
# initialization time. This file contains statements that
#
#   o Control tracing options
#   o Select routing protocols
#   o Manage routing information
#   o Manage independent system routing
#
# See the DIGITAL TCP/IP Services for OpenVMS Management guide for
# instructions on using this file.
#
#_____

interfaces {
    interface all passive ;
};

#
# Protocols:
#
rip on {
    broadcast;
    interface all ripin ripout version 1;
};

#
redirect on;
routerdiscovery server off;
hello off;
```

```
ospf off;
egp off;
bgp off;
snmp off;

#
# Static routes:
#
#static {
#    10.1.2.0 mask 255.255.255.0 gateway 10.1.1.1;
#    default gateway 10.1.2.3;
#    };
#
# Policy:
#
#export proto rip {
#    proto static { all metric 1; };
#    proto direct { all; };
#    proto rip { all; };
#    };
```

Configuring IP Forwarding

- You can manually configure your system as a router.
 - For the current session only:
TCPIP> SET PROTOCOL IP /FORWARD
 - For every time the product starts:
TCPIP> SET CONFIGURATION PROTOCOL IP /FORWARD
- To disable routing:
TCPIP> SET PROTOCOL IP /NOFORWARD
- To disable routing at TCP/IP Services startup:
TCPIP> SET CONFIGURATION PROTOCOL IP /NOFORWARD

Configuring Routing Examples

Configuring Routing

\$ tcpip

TCPIP> **show inter**

Interface	IP_Addr	Network mask	Packets		MTU
			Receive	Send	
WE0	192.206.128.27	255.255.255.0	1	1	1500
LO0	127.0.0.1	255.0.0.0	1624	1624	0

TCPIP>

TCPIP> **show route**

DYNAMIC

Type	Destination	Gateway
DH	127.0.0.1	127.0.0.1
	Loopback address. (The D in the type field seems inappropriate.)	
AN	192.206.128.0/24	192.206.128.27
	Automatically created for configured interfaces.	

TCPIP>

TCPIP> **ping 192.206.126.3**

PING 192.206.126.3 (192.206.126.3): 56 data bytes

ping: sendto:

no route to host

Cannot get to network 192.206.126.

ping: wrote 192.206.126.3 64 chars, ret=-1

(...)

----192.206.126.3 PING Statistics----

2 packets transmitted,

0 packets received,

100% packet loss

TCPIP> **show host**

LOCAL database

Host address	Host name
--------------	-----------

192.206.126.3	ALFMAN, alfman
---------------	----------------

192.206.126.80	COMPCOM
----------------	---------

127.0.0.1	LOCALHOST, localhost
-----------	----------------------

192.206.128.27	den255, DEN255
----------------	----------------

TCPIP>

Adding a Default Route

```
TCPIP> set inter sl0 /host=192.206.127.27 /network_mask=255.255.255.0
      /serial=tta0:
```

Set up the serial line.

```
TCPIP> sho inter
```

Interface	IP_Addr	Network mask	Packets		MTU
			Receive	Send	
WE0	192.206.128.27	255.255.255.0	5	1	1500
SL0	192.206.127.27	255.255.255.0	0	0	1006
LO0	127.0.0.1	255.0.0.0	1756	1756	0

```
TCPIP>
```

```
TCPIP> route add default 192.206.127.27
```

```
add net default: gateway 192.206.127.27
```

Add the default route through the serial line.

```
TCPIP> show route
```

DYNAMIC

Type	Destination	Gateway
DN	<u>0.0.0.0</u>	192.206.127.27
	Default route (0.0.0.0) added.	
AH	127.0.0.1	127.0.0.1
DN	192.206.128.0/24	192.206.128.27

```
TCPIP>
```

```
TCPIP> ping alfman
```

```
PING ALFMAN (192.206.126.3): 56 data bytes
```

Network 192.206.126 is now accessible.

```
64 bytes from 192.206.126.3: icmp_seq=0 ttl=64 time=185 ms
64 bytes from 192.206.126.3: icmp_seq=1 ttl=64 time=183 ms
64 bytes from 192.206.126.3: icmp_seq=2 ttl=64 time=183 ms
64 bytes from 192.206.126.3: icmp_seq=3 ttl=64 time=183 ms
```

```
----ALFMAN PING Statistics----
```

```
4 packets transmitted,
4 packets received,
0% packet loss
round-trip (ms)  min/avg/max = 183/183/185 ms
TCP/IP>
```

Forwarding

TCPIP> **netstat -rn**

Routing tables

Destination	Gateway	Flags	Refs	<u>Use</u>	Interface
-------------	---------	-------	------	------------	-----------

Route Tree for Protocol Family 2

default	192.206.127.27	US	0	<u>18</u>	SL0
---------	----------------	----	---	-----------	-----

Note the Use statistic (18).

127.0.0.1	127.0.0.1	UH	3	184	L00
-----------	-----------	----	---	-----	-----

192.206.128/24	192.206.128.27	U	1	176	WE0
----------------	----------------	---	---	-----	-----

Route Tree for Protocol Family 26

TCPIP> **netstat -rmv**

Shows many characteristics of each route.

Routing tables

Destination	Gateway	Flags	Refs	Use					
Interface	MTU	HopCnt	Expire	RPipe	SPipe	Thresh	RTT	RTTvar	Prec
(...)									

TCPIP> **sysconfig -q inet ipforwarding**

inet:

ipforwarding = 0

Not forwarding yet (ipforwarding = 0).

TCPIP> **set protocol ip/forward**

Enables forwarding of packets.

TCPIP> **sysconfig -q inet ipforwarding**

inet:

ipforwarding = 1

Now the node is a router!

TCPIP> **sysconfig -q inet ipgateway**

inet:

ipgateway = 1

TCPIP>

ROUTED

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

1 - Core environment

Choose Core.

2 - Client components

3 - Server components

4 - Optional components

5 - Shutdown DIGITAL TCP/IP Services for OpenVMS

6 - Startup DIGITAL TCP/IP Services for OpenVMS

7 - Run tests

A - Configure options 1 - 4

[E] - Exit configuration procedure

Enter configuration option: 1

DIGITAL TCP/IP Services for OpenVMS CORE ENVIRONMENT Configuration Menu

Configuration options:

1 - BIND Resolver

2 - Domain

3 - Routing

Choose Routing.

4 - Interfaces

5 - Time Zone

A - Configure options 1 - 5

[E] - Exit menu

Enter configuration option: 3

Dynamic Routing

DYNAMIC ROUTING Configuration

Dynamic routing has not been configured.

You may configure dynamic ROUTED or GATED routing.
You cannot enable both at the same time. If you want to change from one to the other, you must disable the current routing first, then enable the desired routing.

If you enable dynamic ROUTED routing, this host will use the Routing Information Protocol (RIP) - Version 1 to listen for all dynamic routing information coming from other hosts to update its internal routing tables.
It will also supply its own Internet addresses to routing requests made from remote hosts.

If you enable dynamic GATED routing, you will be able to configure this host to use any combination of the following routing protocols to exchange dynamic routing information with other hosts on the network:

- Routing Information Protocol (RIP) - Version 1 & 2
- Router Discovery Protocol (RDISC)
- Open Shortest Path First (OSPF)
- Exterior Gateway Protocol (EGP)
- Border Gateway Protocol (BGP-4)
- Static routes

* Do you want to configure dynamic ROUTED or GATED routing [NO]: y

* Do you want to enable GATED routing configuration [NO]:

ROUTED option

If you enable the 'supply' option of dynamic routing, this host will supply dynamic routing information to other hosts on the network whether it is acting as an internetwork gateway or not.

* Do you want this host to supply its dynamic routing information [NO]: y

There can be one and only one host on your network which defines a default network route.

If you want this host to supply the default network route, then all packets which cannot be routed to their destination host or network will be routed via this host.

* Do you want this host to supply the default network route [NO]: y

(...)

Restart the network and ROUTED should be running.

\$

\$ **pipe sho sys | search sys\$input routed**

000000CB TCPIP\$ROUTED (...)

\$

\$ **sho proc/id=cb /all**

22-JUN-1999 17:03:03.32 User: INTERNET

Process ID: 000000CB

Node: DEN255

Process name: "TCPIP\$ROUTED"

Eavesdropping on the ROUTED process.

Terminal:

User Identifier: [SYSTEM]

Base priority: 4

Default file spec: Not available

Number of Kthreads: 1

Devices allocated: BG255:

BG257:

Note the BG devices.

Process Quotas:

(...)

There are 2 processes in this job:

TCPIP\$INET_ACP

TCPIP\$ROUTED (*)

\$

\$

TCPIP>

TCPIP> **set route 192.206.126.0 /gate=192.206.127.27 /network**

Add a new route to network 192.206.126 through 192.206.127.27.

TCPIP>

TCPIP>

TCPIP> **sho route**

DYNAMIC

Type	Destination	Gateway
DH	127.0.0.1	127.0.0.1
DN	<u>192.206.126.0/24</u>	<u>192.206.127.27</u>
AN	192.206.128.0/24	192.206.128.27

TCPIP>

TCPIP>

TCPIP> **route add default 192.206.127.27**

add net default: gateway 192.206.127.27

TCPIP>

TCPIP>

TCPIP> **sho route**

DYNAMIC

Type	Destination	Gateway
DN	<u>0.0.0.0</u>	192.206.127.27

Show the default route.

DH	127.0.0.1	127.0.0.1
AN	192.206.126.0/24	192.206.127.27
DN	192.206.128.0/24	192.206.128.27

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$

Tracing ROUTED Traffic

\$

\$ tcpip

TCPIP> show device_socket bg255

		Port		Remote	
Device_socket	Type	Local	Remote	Service	Host
bg255	DGRAM	<u>520</u>	0		0.0.0.0

ROUTED uses port 520.

TCPIP>

TCPIP>

TCPIP> show device_socket bg255/full

Device_socket: bg255	Type: DGRAM	LOCAL	REMOTE
	Port:	<u>520</u>	0
	Host:	0.0.0.0	0.0.0.0
	Service:		
	(...)		

\$

\$

Tracing ROUTED Communications

\$ **tcptrace/port=local=520**

TCPIPtrace RCV packet 1 at 22-JUN-1999 17:38:23.07

Reports on network traffic involving local port 520 (used by dynamic routing daemons).

	IP Address	Port
Source	192.206.126.3	<u>520</u>
Destination	192.206.126. <u>255</u>	<u>520</u>

Note the destination address is a broadcast address.

			Packet Length	72		
UDP						
			Hex	Count	Ascii	
-----	-----	-----	-----	----	-----	
037ECEC0	6AF1111E	00009B2C	48000045	0000	E..H,.....j..~.	
00000102	2DFC3400	08020802	FF7ECEC0	0010	..~.....4.-....	
00000000	00000000	037FCEC0	00000200	0020		
00000000	007ECEC0	00000200	01000000	0030	~.....	
		01000000	00000000	0040		

TCPIPtrace RCV packet 8 at 22-JUN-1999 17:38:53.07

	IP Address	Port
Source	192.206.126.3	520
Destination	192.206.126.255	520

Packet Length 72

UDP						
			Hex	Count	Ascii	
-----	-----	-----	-----	----	-----	
037ECEC0	68F1111E	00009D2C	48000045	0000	E..H,.....h..~.	
00000102	2DFC3400	08020802	FF7ECEC0	0010	..~.....4.-....	
00000000	00000000	037FCEC0	00000200	0020		
00000000	007ECEC0	00000200	01000000	0030	~.....	
		01000000	00000000	0040		

Note the time differential between trace entries.

Fully Tracing ROUTED Traffic

\$ tcptrace/port=local=520/full

 TCPIPtrace full display RCV packet 5 at 22-JUN-1999 17:42:23.05

← IP header information.

IP Version = 4, IHL = 5, TOS = 00, Total Length = 72 = ^x0048

IP Identifier = ^x2CB2, Flags (0=0,DF=0,MF=0),

Fragment Offset = 0 = ^x0000, Calculated Offset = 0 = ^x0000

IP TTL = 30 = ^x1E, Protocol = 17 = ^x11, Header Checksum = ^xF153

IP Source Address = 192.206.126.3

IP Destination Address = 192.206.126.255

UDP Source Port = 520, UDP Destination Port = 520

UDP Header and Datagram Length = 52 = ^x0034, Checksum = ^xFC2D

037ECEC0	53F1111E	0000B22C	48000045	0000	E..H,.....S..~.
00000102	2DFC3400	08020802	FF7ECEC0	0010	..~.....4.-....
00000000	00000000	037FCEC0	00000200	0020	
00000000	007ECEC0	00000200	01000000	0030	~.....
		01000000	00000000	0040	

Displays more header information (/FULL).

\$

Introduction

The focus in this module is on the Domain Name Service. TCP/IP Services for OpenVMS uses version 8.1.2 of BIND (Berkeley Internet Name Domain). The concepts and brief history of BIND are presented to help you understand how the software evolved. The resolver side (client side) is examined and an example is used to solidify the concepts. The same approach is used with the BIND server.

Objectives

Upon completion of this module, you should be able to:

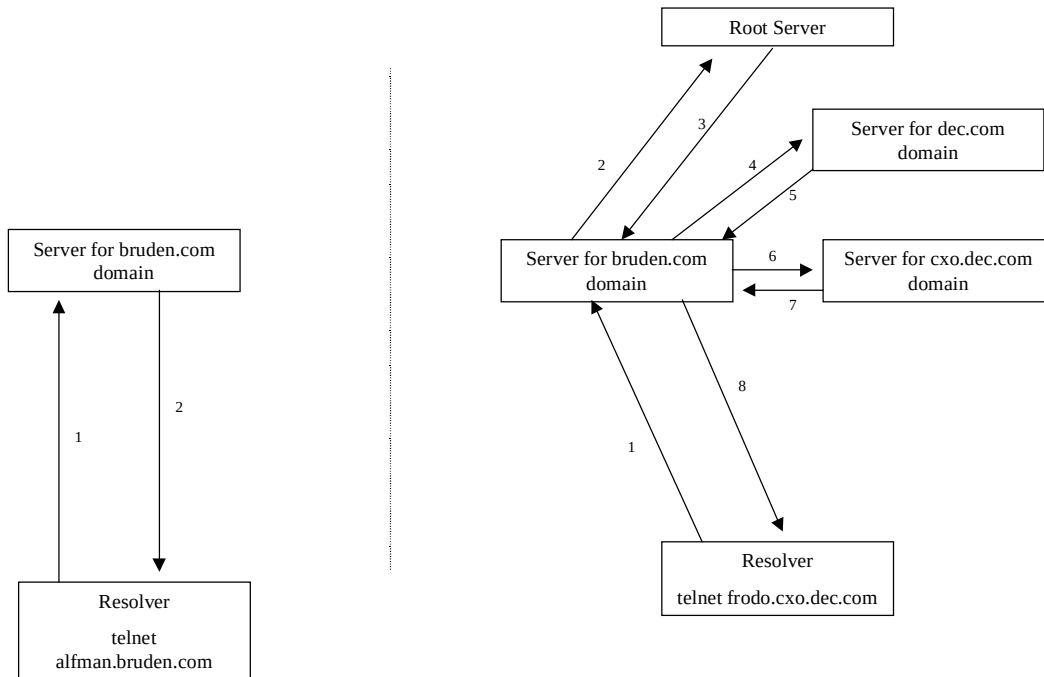
- Discuss the history of BIND.
- Describe BIND at a conceptual level.
- Configure the resolver software.
- Configure the server software.
- Discuss the load broker and metric server processes.

Topics

- History
- Concepts
- Resolver
- Server
- NSLOOKUP
- Load Broker and Metric Server

Domain Name Service (DNS)

- It is easier to use (and remember) a node name than an address.
- The primary purpose of the DNS service is to translate node names into IP addresses. DNS also translates addresses back into node names.
- Client systems, called *resolvers*, request translations from name server systems.
- Servers pass the request to other servers until the name is resolved to an address.



Servers Translating Node Names

DNS History

Early Internet

- Without DNS, Internet nodes would be responsible for tracking and storing tens of thousands, or millions, of node names and their associated IP addresses.
- During the 1970s, the Internet was called the ARPANET, and it had several hundred nodes.
- All node names and IP addresses were stored in one file (HOSTS.TXT) on one node (SRI_NIC) at the Stanford Research Institute in Menlo Park, CA.
 - Other nodes would FTP this file and change it into the form needed for their particular operating system.
 - As the number of nodes grew, the performance impact on SRI-NIC was massive.
 - Every addition to HOSTS.TXT required administrative intervention by someone at Stanford.
 - Every additional node meant another node doing periodic ftp requests, which further exacerbated the problem.
 - A node wanting to keep up to date would potentially have to ftp many times a day.
- Node names were hard to manage. Duplicate names could break the whole system.
- The problems were:
 - Node name uniqueness.
 - Centralized administrative nightmare.
 - Keeping up to date with node names and addresses.

JEEVES and BIND

- The first DNS was developed in 1984 by Paul Mockapetris of USC.
 - His solution included a hierarchical name space with distributed administrative authority.
 - Addresses would be served out to the clients (or other servers) that requested the information.
 - Because his solution involved serving to those who wanted a service, he called it JEEVES (a cartoon butler's name).
- Most of today's DNS implementations use the Berkeley Internet Name Domain (BIND) software.
 - BIND was written by Kevin Dunlap and is maintained by Paul Vixie of the Internet Software Consortium (ISC).
 - TCP/IP Services for OpenVMS uses BIND version 8.1.2.
 - Many other TCP/IP implementations still use BIND version 4.9 (or close to it).

Hierarchical Name Format

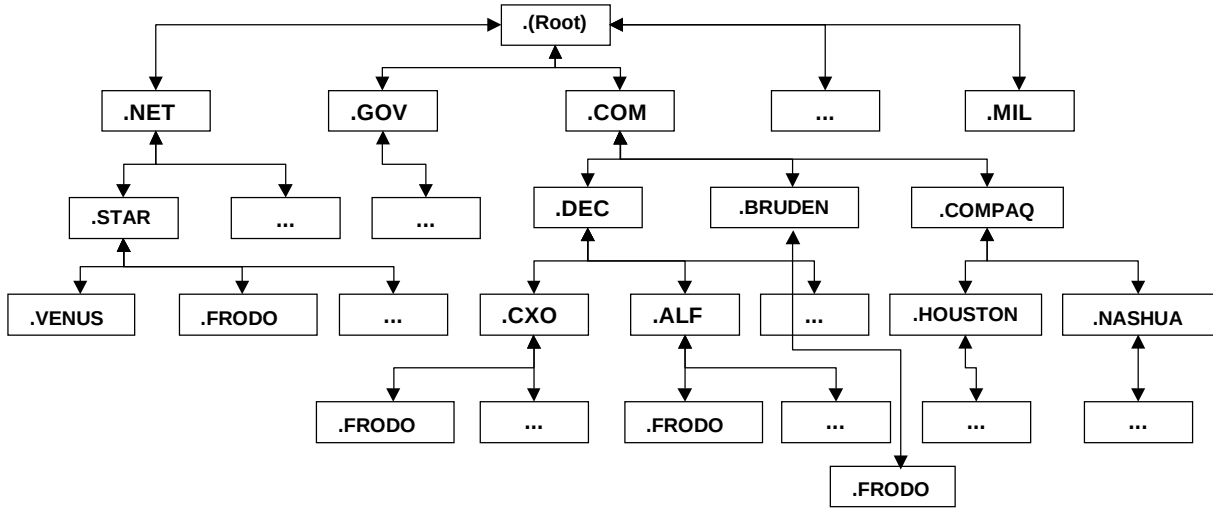
- The most recognizable part of BIND is the hierarchical name format: `alfman.bruden.com`, or `frodo.alf.dec.com`.
 - BIND groups the names into domains.
 - Names start with specific information (`frodo`) and end with broad information (`com`).
- The top of the hierarchy is the dot (`.`), which officially terminates the name `frodo.alf.dec.com`.

Common top-level domains include:

- COM — commercial
 - EDU — educational
 - GOV — government
 - MIL — military
 - NET — network organizations
 - ORG — nonprofit and miscellaneous organizations
- All top-level domains and the next level down are under the control of the Internet Corporation for Assigned Names and Numbers (ICANN, formerly InterNIC). ICANN is a not-for-profit organization recently appointed by the U.S. Government Commerce Department to oversee registration of Internet names.
 - NSI (Network Solutions), through which most name requests flowed, is balking at relinquishing control to ICANN (as of 6/25/1999). A political and legal battle looms and threatens the integrity of the DNS as this is being written.
 - Names are rented for \$50 per year (for example, `bruden.com`).

Domains and Subdomains

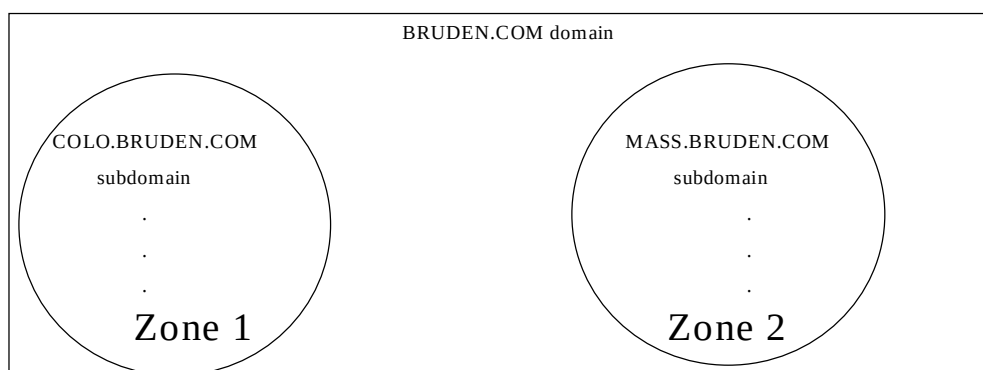
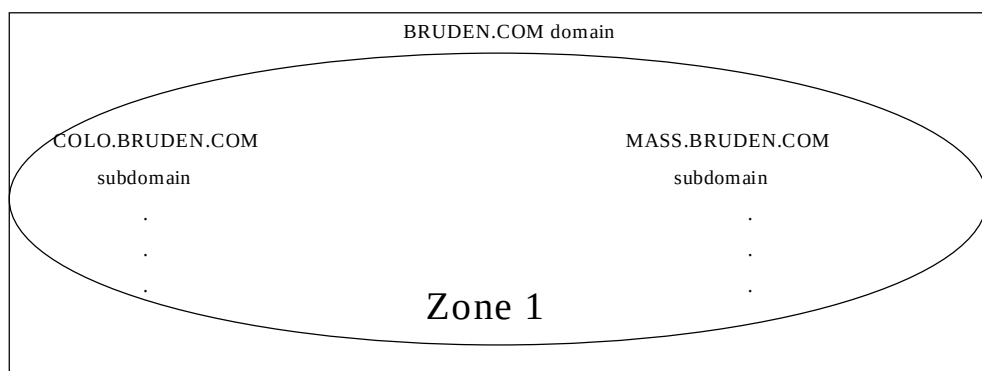
- Names underneath the first two levels are the administrative responsibility of the company or organization, thus decentralizing the administrative responsibilities.
 - You can organize these levels into subdomains.
 - You can further divide each subdomain if necessary.
 - Only the top-level domain names (and the root) are the responsibility of the current DNS organizing entity.
 - This solves the problems mentioned earlier.
- A small company may have just one domain (bruden.com), while a large company may have one domain and hundreds of subdomains (such as alf.dec.com or cxo.dec.com).
 - Each subdomain can be administered separately, or an administrator can be responsible for several subdomains.
 - Names can be duplicated within different domains.



Portion of Internet BIND Domain

DNS Zones

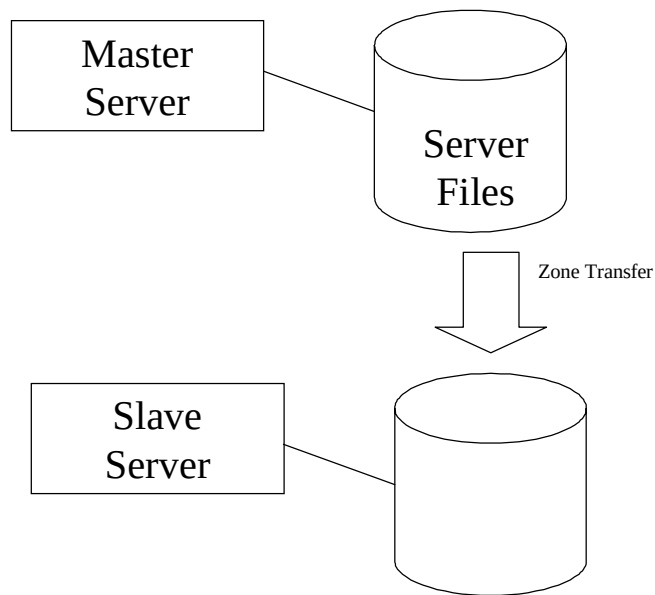
- Each server node can serve out information about several subdomains or about one subdomain.
- The part of the name hierarchy over which a server has authoritative name information is called a *zone*.
 - A zone may be exactly the same as a subdomain (one server for each subdomain), or a zone could encompass several subdomains (one server handling several subdomains).
 - A zone could also be a portion of a domain or a subdomain.
- The following figure shows BRUDEN.COM as a single zone and as multiple zones.



Single and Multiple DNS Zones

DNS Master and Slave Servers

- Servers can be *master* (formerly called primary) or *slave* (formerly called secondary).
- The master server is authoritative for a zone and gets its data from disk files on the server node.
- The slave server gets its data from the primary server through *zone transfers*.
- The slave server can keep some local data on disk in case the master is unavailable.
- The slave server is also considered authoritative for a zone.



Relationship of Master Server to Slave Server

Files and Resource Records

- Server zone information is kept in files and organized into resource records (RRs).
 - **Forward translation file** (bruden_com.db)
Maps host names to IP addresses.
 - **Reverse translation file** (128_206_192_in-addr_arpa.db)
Maps an address back to a host name.
 - **Local loopback forward translation file** (LOCALHOST.DB)
Translates the heavily used LOCALHOST node name to the address 127.0.0.1 without extra overhead.
 - **Reverse loopback translation file** (127_0_0.DB)
Reverses the loopback address translation without extra network overhead.
 - **Root name server file** (ROOT.HINT)
Contains names and addresses of the servers handling the top of the Internet domain hierarchy (or local hierarchy if intranet).
 - **Configuration file** (TCPIP\$BIND.CONF)
Contains statements that name all of the above files and allows the administrator to assign characteristics specific to this server.
- The master server would have all of these files.
- The slave server must have configuration, hints, and loopback files, and could have forward and reverse translation files.
- Files are in SYS\$SPECIFIC:[TCPIP\$BIND].

All zone files use the same types of records.

Standard Resource Record Types	Description	Function
SOA	Start of authority	Marks the beginning of a zone's data and defines parameters that affect the entire zone.
NS	Name server	Identifies the name server of a domain.
A	Address	Maps a host name to an address.
PTR	Pointer	Maps an address to a host name.
MX	Mail exchange	Identifies where to deliver mail for a given domain.
CNAME	Canonical name	Associates an alias host name.
HINFO	Host information	Describes the hardware and system of a host.
WKS	Well-known service	Advertises network services.

DNS record format:

[name] [ttl] IN type data

■ **name**

Specifies the name of the domain referenced by a resource record. The string entered for name is the current domain unless it ends with a dot. Blank indicates that the record applies to the domain last named.

■ **ttl**

Time, in seconds, that the information in this resource record should be kept in cache. Usually, the time-to-live field is left blank, and the default ttl, set for the entire zone in the SOA record, is used.

■ **IN**

Identifies the record as an Internet DNS resource record (not Hesiod or CHAOS).

■ **type**

Identifies what type of resource record this is.

■ **data**

Information specific to this type of resource record. In an **A** record, this is the field that contains the actual IP address.

For example:

- Address record
den255.bruden.com. 12H IN A 192.206.128.27.
- Name server record
bruden.com. 12H IN NS den255.bruden.com.
- Start of authority record (SOA)

Usually preceded by the \$ORIGIN entry, which is appended to any name not ending in a dot. Using the \$ORIGIN entry, the first field (name) of the address record above could have been just **den255**. It would also replace the @ symbol in the following SOA record.

\$ORIGIN bruden.com.

```
@      IN      SOA      den255.bruden.com.      system.bruden.com.
{
```

```
1          ; Serial Number
          ; Increase each time this file is changed.
600        ; Refresh Time
          ; Zone will be refreshed every 10 mins.
          ; Tells the slave how often to check to see if its serial.
; number is less than the one in the master's SOA record.
300        ; Retry
          ; A failed refresh should be retried every 5 mins.
          ; Slave tries to reconnect to master every 'retry' seconds.
172800 ; Expire
          ; Time after which zone is no longer authoritative.
          ; Slave stops giving out answers after 'expire' seconds of
          ; silence from master.
43200}     ; Minimum TTL
          ; Minimum time to live for exported information.
          ; Slaves will cache data for at least this long.
```

- PTR record
PTR records are for reverse address translations (address to name).
192.206.126.3. IN PTR alfman.bruden.com

DNS Server Types

Master Server

The master server has zone data in files and reads the data at startup.

- Make changes to files in the master server's files.
- Increment the serial number in the SOA record with each change made to the file (like an OpenVMS file version number).
- The master server is authoritative for its zone.

Slave Server

The slave server gets data from the master server through zone transfer.

- The slave server is called the *secondary master* in BIND v4.
- It also reads from local files (if configured) and checks with the master to see if data is accurate.
- Files have start of authority (SOA) records containing a serial number.
- If the slave sees that its serial number is less than the master's serial number, a zone transfer starts.
- Slaves provide redundancy, server load spreading, and locality of server.
- If the master is down, the slave's local database file information is served.
- The slave server is authoritative for its zone.

Caching-Only Server

- Caching-only is a less complex server with fewer data files.
- Use it to spread the server processing load over multiple server nodes.
 - It requests items of information (not a zone transfer) from master or slave servers when first queried by a resolver.
 - Subsequent queries are handled from the cache of the caching-only server, which develops over a period of time.
- A hints file and a loopback file are required.
- The caching-only server is considered *nonauthoritative*.

Forwarder Server

- The forwarder server is configured like a standard master or slave server.
 - Other servers point to the forwarder when a query involves servers beyond the company's domain.
 - The forwarder minimizes queries to the top-level name servers by building up a rich cache of information that it gets from the top-level servers.
- The other servers, pointing to the forwarder, give the forwarder characteristic to a particular server.
 - No special syntax or set-up on the forwarder is needed.
 - There is a forwarders option in the configuration file for all the other servers.

DNS Query Types

Recursive

- Resolvers typically send recursive queries.
 - The response must be the requested data, or an error.
 - The response does **not** refer the resolver to another name server.
- Servers typically send *iterative* queries between each other, which may produce a response identifying a better server to use.

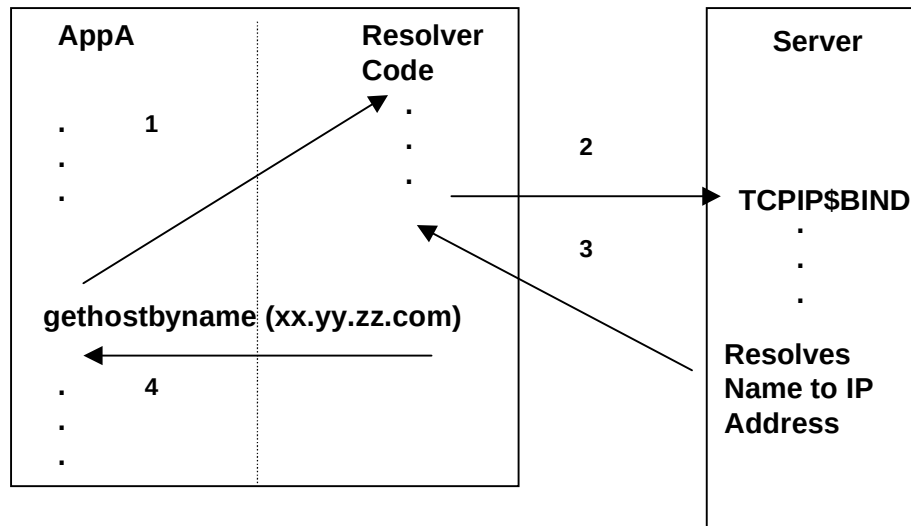
Iterative

Servers querying other servers typically use iterative queries.

- The response is the best answer already known by the queried server.
- Iterative queries do not force the queried server to begin querying other servers.

DNS Daemon Processes

- The server executes a daemon process, `TCPIP$BIND`, to handle the requests for address information.
- Resolvers (clients) run various applications that call library routines (such as `gethostbyname( )`) designed to request information from the server if necessary.
- Both clients and servers run the resolver code.
- In the following figure, the client runs resolver code, which contacts the server for translation.



DNS Daemon Processes on the Client and Server

Resolver Example

\$ **tcpip**

TCP/IP> **show name**

BIND Resolver Parameters

Local domain: Not defined

System

Changes can be systemwide.

State: Started, Disabled

Name service not enabled yet.

Transport: UDP

Domain:

Retry: Not defined

Timeout: Not defined

Servers: No values defined

Path: No values defined

Process

Changes can be per process.

State: Disabled

Transport:

Domain:

Retry:

Timeout:

Servers:

Path:

TCP/IP>

TCP/IP>

TCP/IP> **exit**

\$

\$ **tcpip**

TCPIP>

TCPIP>

TCPIP>

TCPIP> **sho route**

Type	Destination	<u>DYNAMIC</u>	Gateway
DN	0.0.0.0		192.206.127.27
AH	127.0.0.1		127.0.0.1
DN	192.206.128.0/24		192.206.128.27

TCPIP>

TCPIP>

TCPIP> **ping compcom**

Using TCPIP\$HOSTS to get the address.

PING COMPCOM (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=127 time=186 ms

64 bytes from 192.206.126.80: icmp_seq=1 ttl=127 time=184 ms

64 bytes from 192.206.126.80: icmp_seq=2 ttl=127 time=185 ms

----COMPCOM PING Statistics----

4 packets transmitted,

3 packets received,

25% packet loss

round-trip (ms) min/avg/max = 184/185/186 ms

TCPIP>

TCPIP>

TCPIP> **sho host compcom**

LOCAL database

From LOCAL database.

Host address	Host name
--------------	-----------

192.206.126.80	COMPCOM
----------------	---------

TCPIP>

TCPIP>

TCPIP> **set nohost compcom**

Remove node compcom from the local database.

TCPIP>

TCPIP> **sho host compcom**

%TCPIP-E-HOSTERROR, cannot process host request

-TCPIP-W-NORECORD, information not found

Not there.

-RMS-E-RNF, record not found

TCPIP>

TCPIP>

TCPIP> **ping compcom**

ping: unknown host compcom

Cannot do, ping by name fails.

TCPIP>

TCPIP>

TCPIP> **ping 192.206.126.80**

Using the address to ping works fine.

PING 192.206.126.80 (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=127 time=187 ms

64 bytes from 192.206.126.80: icmp_seq=1 ttl=127 time=184 ms

64 bytes from 192.206.126.80: icmp_seq=2 ttl=127 time=184 ms

----192.206.126.80 PING Statistics----

3 packets transmitted,

3 packets received,

0% packet loss

round-trip (ms) min/avg/max = 184/185/187 ms

TCPIP>

TCPIP>

\$

Resolver Configuration

\$

\$ @sys\$startup:tcpip\$conf

Configure the resolver.

TCP/IP Network Configuration Procedure

(...)

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
Select core.
- 2 - Client components
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: 1

DIGITAL TCP/IP Services for OpenVMS CORE ENVIRONMENT Configuration Menu

Configuration options:

- 1 - BIND Resolver
Select BIND resolver.
- 2 - Domain
- 3 - Routing
- 4 - Interfaces
- 5 - Time Zone

- A - Configure options 1 - 5
- [E] - Exit menu

Enter configuration option: 1

BIND RESOLVER Configuration

A BIND resolver has not been configured.

DIGITAL TCP/IP Services for OpenVMS supports the Berkeley Internet Name Domain (BIND) resolver. BIND is a network service that enables clients to name resources or objects and share information with other objects on the network.

Before configuring your system as a BIND resolver, you should first be sure that there is at least one system on the network configured as either a BIND primary or secondary server for this domain.

Note old terminology: primary (master) and secondary (slave). You can specify a BIND server by its address or name; however, if specified by name, an entry for it must exist in the TCPIP\$HOST database.

You will be asked one question for each server.
Press Return at the prompt to terminate the list.

Enter your BIND server name: alfman

Use just one server.

Enter your BIND server name:

You could set up others.

You must provide the domain name in order to run the BIND resolver. This information should be available from your Internet network administrator.

You will now be asked to configure the domain name.

DOMAIN Configuration

Enter Internet domain: bruden.com

Domain name.

Stop and restart the network.

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

A - Configure options 1 - 4

[E] - Exit configuration procedure

Enter configuration option: **5**

Begin Shutdown...

%JBC-E-JOBQUEDIS, system job queue manager is not running

NFSS\$SERVER* not found

TCPIP\$NTP server 000003C9 stopped

TCPIP\$ESNMP stopped

TCPIP\$OS_MIBS stopped

TCPIP\$HR_MIB stopped

TCPIP\$METRIC server 000003CA stopped

TCPIP\$FTP shutdown complete

The following Internet devices are still active
on your system:

BG50:

Note the following:

Before restarting the Internet software you
should stop the processes associated with the
above BG devices. To locate these processes,
issue the DCL command SHOW DEVICE BGn /FULL.

Shutdown request completed.

Press Return to continue ...

(...)

\$

\$ sho dev bg50/full

Examine BG50, note the PID.

Device BG50:, device type unknown, is online, record-oriented device, network device, mailbox device.

Error count	0	Operations completed	3
Owner process	"_FTA1:"	Owner UIC	[SYSTEM]
Owner process ID	<u>000000D7</u>	Dev Prot	S:RWPL, O:RWPL, G:RWPL, W:RWPL
Reference count	1	Default buffer size	256

\$

\$

\$ stop/id=d7

Remove PID hex D7. (First check to see what PID 0xD7 is doing!)

\$

\$ sho dev bg50/full

Frees up the device.

%SYSTEM-W-NOSUCHDEV, no such device available

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests
- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **6**

Restart the network.

Begin Startup...

(...)

%TCPIP-I-STARTNAME, starting name service

(...)

\$

\$ **tcpip**

TCPIP> **sho name**

BIND Resolver Parameters

Local domain: bruden.com

The domain name is set.

System

State: Started, Enabled

Software is enabled.

Transport: UDP

Domain: bruden.com

Retry: 4

Timeout: 4

Servers: ALFMAN

Alfman is the server.

Path: No values defined

Process

State: Enabled

Transport:

Domain:

Retry:

Timeout:

Servers:

Path:

TCPIP>

TCPIP>

TCPIP>

TCPIP> **sho host compcom**

 BIND database

 Address has been served from alfman.

Server: 192.206.126.3 ALFMAN

Host address Host name

192.206.126.80 COMPCOM.bruden.com

TCPIP>

TCPIP>

TCPIP> **ping compcom**

PING compcom.bruden.com (192.206.126.80): 56 data bytes

64 bytes from 192.206.126.80: icmp_seq=0 ttl=127 time=185 ms

64 bytes from 192.206.126.80: icmp_seq=1 ttl=127 time=184 ms

64 bytes from 192.206.126.80: icmp_seq=2 ttl=127 time=184 ms

----compcom.bruden.com PING Statistics----

4 packets transmitted,

3 packets received,

25% packet loss

round-trip (ms) min/avg/max = 184/184/185 ms

TCPIP>

TCPIP>

TCPIP> **sho host/local compcom**

 Not found locally.

%TCPIP-E-HOSTERROR, cannot process host request

-TCPIP-W-NORECORD, information not found

-RMS-E-RNF, record not found

TCPIP>

TCPIP>

TCPIP> **sho host /server=alfman**

What does alfman know?

BIND database

Server: 192.206.126.3 ALFMAN

Host address	Host name
--------------	-----------

192.206.128.27	DEN255.bruden.com
192.206.126.3	alfman.bruden.com
192.206.127.27	den255-sl.bruden.com
192.206.127.3	alfman-sl.bruden.com
192.206.126.80	compcom.bruden.com
127.0.0.1	localhost.bruden.com
192.206.126.4	cuff.bruden.com

TCPIP>

TCPIP>

TCPIP> **sho host /local**

What do we know locally?

LOCAL database

Host address	Host name
--------------	-----------

192.206.126.3	ALFMAN, alfman
192.206.127.3	ALFMAN-SL
127.0.0.1	LOCALHOST, localhost
192.206.128.27	den255, DEN255

TCPIP>

TCPIP>

TCPIP> **exit**

NSLOOKUP

- BIND provides a query tool, nslookup, which interactively queries domain name servers (BIND servers).
- You can use nslookup to look at DNS records.

\$

\$ nslookup:= \$sys\$system:tcpip\$nslookup.exe

Set up a foreign command for the nslookup program.

\$ nslookup

Default Server: alfman.bruden.com

Address:

192.206.126.3

>

> compcom

Requests the address for compcom.

Server: alfman.bruden.com

Address:

192.206.126.3

Name: compcom.bruden.com

Address:

192.206.126.80

Receives the address.

> cuff

Server: alfman.bruden.com

Address:

192.206.126.3

Name: cuff.bruden.com

Address:

192.206.126.4

>

NSLOOKUP Internal Commands

■ LS

- Lists the information available for the domain specified.
- The default output contains host names and their Internet addresses.
- Qualifiers include the following:

Qualifier	Description
-a	Lists canonical names and aliases in the domain.
-d	Lists all entries in the domain.
-h	Lists CPU and operating system information for the domain, HINFO entries.
-m	Lists mail exchangers in the domain, MX entries.
-s	Lists well-known services in the domain, WKS entries.
-t	Lists information of a specified type; the default is the default querytype.
> file	Directs the output to the specified file.
>> file	Appends the output to the specified file.

■ LSERVER

- Changes the default server to the specified server.
- If necessary, this command uses the **initial** default server to translate the specified server name into an address.

■ SERVER

- Changes the default server to the specified server.
- If necessary, this command uses the **current** default server to translate the specified server name into an address.

NSLOOKUP Examples

```
> ls -a bruden.com
```

```
[alfman.bruden.com]
```

Listing names and aliases.

```
$ORIGIN bruden.com.
```

```
hosts-0          12H IN CNAME localhost
hosts-1          12H IN CNAME alfman
hosts-2          12H IN CNAME DEN255
hosts-3          12H IN CNAME cuff
hosts-4          12H IN CNAME alfman-sl
hosts-5          12H IN CNAME den255-sl
hosts-6          12H IN CNAME compcom
```

```
>
```

```
> ls -d bruden.com
```

```
[alfman.bruden.com]
```

Listing all entries in the domain.

```
@          12H IN SOA  alfman postmaster.alfman (
                        1          ; serial
                        5M          ; refresh
                        1M          ; retry
                        2W          ; expiry
                        12H )       ; minimum
```

```

                                12H IN NS    alfman
DEN255                        12H IN A      192.206.128.27
hosts-0                       12H IN CNAME localhost
hosts-1                       12H IN CNAME alfman
alfman                        12H IN A      192.206.126.3
den255-sl                    12H IN A      192.206.127.27
hosts-2                       12H IN CNAME DEN255
alfman-sl                    12H IN A      192.206.127.3
hosts-3                       12H IN CNAME cuff
compcom                      12H IN A      192.206.126.80
hosts-4                       12H IN CNAME alfman-sl
hosts-5                       12H IN CNAME den255-sl
localhost                    12H IN A      127.0.0.1
hosts-6                       12H IN CNAME compcom
cuff                          12H IN A      192.206.126.4
@                              12H IN SOA   alfman postmaster.alfman (
                                1           ; serial
                                5M          ; refresh
                                1M          ; retry
                                2W          ; expiry
                                12H )       ; minimum
```

>

> **ls -t a bruden.com > nslook.out**

The file contains an output listing records of type A (-t a).

[alfman.bruden.com]

Received 24 answers (0 records).

> **exit**

\$

\$

\$ **type nslook.out**

> **ls -t a bruden.com**

[alfman.bruden.com]

DEN255	12H IN A	192.206.128.27
alfman	12H IN A	192.206.126.3
den255-sl	12H IN A	192.206.127.27
alfman-sl	12H IN A	192.206.127.3
compcom	12H IN A	192.206.126.80
localhost	12H IN A	127.0.0.1
cuff	12H IN A	192.206.126.4

\$

BIND Master Server Setup

TCPIP\$CONFIG

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

1 - Core environment

Select Core.

2 - Client components

(...)

Enter configuration option: **1**

DIGITAL TCP/IP Services for OpenVMS CORE ENVIRONMENT Configuration Menu

Configuration options:

1 - BIND Resolver

2 - Domain

Select Domain.

3 - Routing

4 - Interfaces

5 - Time Zone

A - Configure options 1 - 5

[E] - Exit menu

Enter configuration option: **2**

DOMAIN Configuration

Enter Internet domain [bruden.com]:

Enable BIND

DIGITAL TCP/IP Services for OpenVMS Configuration Menu

Configuration options:

- 1 - Core environment
- 2 - Client components
- 3 - Server components

Select Server components.

(...)

Enter configuration option: **3**

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Disabled

Must be enabled.

(...)

- 16 - LBROKER Enabled
- 17 - DHCP Disabled
- A - Configure options 1 - 17
- [E] - Exit menu

Enter configuration option: **1**

BIND SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

BIND SERVER configuration options:

1 - Enable service on this node

[E] - Exit BIND configuration

Enter configuration option: **1**

The TCPIP\$AUX identifier already exists with value 3655

Creating BIND Service Entry

Checking for BIND template files in SYS\$SPECIFIC:[TCPIP\$BIND]

Creating template SYS\$SPECIFIC:[TCPIP\$BIND]TCPIP\$BIND_CONF.TEMPLATE

Creating file SYS\$SPECIFIC:[TCPIP\$BIND]127_0_0.DB

Creating file SYS\$SPECIFIC:[TCPIP\$BIND]LOCALHOST.DB

Creating file SYS\$SPECIFIC:[TCPIP\$BIND]ROOT.HINT

Note the files created — no address translation files and only a template configuration file.

Configuration Files

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Enabled
BIND is configured now (partially).
- 2 - BOOTP Disabled

(...)

Shut down and restart the network.

%TCPIP-I-ENABSERV, enabling services

%TCPIP-I-STARTNAME, starting name service

(...)

\$

\$ **sho sys**

OpenVMS V7.2 on node DEN255 24-JUN-1999 08:15:28.74 Uptime 0 01:05:23

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
00000081	SWAPPER	HIB	16	0	0 00:00:03.12	0	0
00000102	TCPIP\$OS_MIBS	LEF	8	202	0 00:00:00.20	265	151

(...)

Shows no TCPIP\$BIND process!

\$ **set def sys\$specific:[tcpip\$bind]**

\$

\$ **dir/size/date**

Directory SYS\$SPECIFIC:[TCPIP\$BIND]

Contains BIND support and configuration files.

127_0_0.DB;1	2	24-JUN-1999 08:04:16.06
LOCALHOST.DB;1	2	24-JUN-1999 08:04:16.26
LOGIN.COM;1	1	24-JUN-1999 08:04:15.05

```

ROOT.HINT;1          5  24-JUN-1999 08:04:16.41
TCPIP$BIND_CONF.TEMPLATE;1 4  24-JUN-1999 08:04:15.86
TCPIP$BIND_RUN.LOG;1    2  24-JUN-1999 08:05:40.80

```

Total of 6 files, 16 blocks.

\$

\$ **sho time**

24-JUN-1999 08:16:36

\$

\$ **type TCPIP\$BIND_RUN.LOG;1**

Check the BIND log file to help figure this out.

\$ SET NOON

\$ VERIFY = F\$VERIFY(F\$TRNLNM("SYLOGIN_VERIFY"))

Thu 24 08:05:41 NOTICE: starting. Digital TCP/IP Services for OpenVMS - BIND
8.1.2 V5.0-9

Thu 24 08:05:41 CRITICAL: **can't open 'tcpip\$bind_server_data:tcpip\$bind.conf'**

This file does not exist.

Thu 24 08:05:41 CRITICAL: can't open 'tcpip\$bind_server_data:tcpip\$bind.conf'
can't open 'tcpip\$bind_server_data:tcpip\$bind.conf'

TCPIP\$BIND job terminated at 24-JUN-1999 08:05:41.56

Accounting information:

Buffered I/O count:	85	Peak working set size:	3616
Direct I/O count:	37	Peak virtual size:	170576
Page faults:	725	Mounted volumes:	0
Charged CPU time:	0 00:00:00.31	Elapsed time:	0 00:00:00.82

\$

You must set up the configuration files before starting the BIND server.

\$ **dir *.conf**

Where is TCPIP\$BIND.CONF ?

%DIRECT-W-NOFILES, no files found

\$

TCPIP\$BIND.CONF

\$ dir

Directory SYS\$SPECIFIC:[TCPIP\$BIND]

```
127_0_0.DB;1          LOCALHOST.DB;1          LOGIN.COM;1          ROOT.HINT;1
TCPIP$BIND_CONF.TEMPLATE;1      TCPIP$BIND_RUN.LOG;1
```

Total of 6 files.

\$

\$ copy TCPIP\$BIND_CONF.TEMPLATE;1 tcpip\$bind.conf

Make your own .CONF file.

\$ type tcpip\$bind.conf

```
//
//  TCPIP$BIND_CONF.TEMPLATE - Sample BIND server configuration file
//                               (BIND Version 8.1 or later)
//
//                               IMPORTANT
//  This file contains information that is read by the BIND server at
//  initialization time. You must edit this file and save it to
//      SYS$SPECIFIC:[TCPIP$BIND]TCPIP$BIND.CONF
//  before starting the BIND server.
//
//_____
options {
```

```
directory "sys$specific:[tcpip$bind]";
    named-xfer "tcpip$bind_server_xfer";
    check-names master warn;          /* default. */
    datasize 20M;
    transfers-in 6;
    max-transfer-time-in 30;          // number of minutes an inbound zone
                                      // transfer may run.
    recursion yes;
};

zone "bruden.com" in {
    type master;
    file "bruden_com.db";
    allow-update { none; };
    allow-transfer { any; };
};
```

Defines the bruden.com zone.

```
zone "126.206.192.in-addr.arpa" in {
    type master;
    file "126_206_192_in-addr_arpa.db";
};
zone "127.206.192.in-addr.arpa" in {
    type master;
    file "127_206_192_in-addr_arpa.db";
};
zone "128.206.192.in-addr.arpa" in {
    type master;
    file "128_206_192_in-addr_arpa.db";
};
```

The above files define the reverse of the bruden.com zone.

(...)

```
zone "localhost" IN {
    type master;
    file "localhost.db";
    check-names fail;
    allow-update { none; };
    allow-transfer { any; };
};
zone "0.0.127.in-addr.arpa" IN {
    type master;
    file "127_0_0.db";
    check-names fail;
    allow-update { none; };
    allow-transfer { any; };
};
zone "." IN {
    type hint;
    file "root.hint";
};
```


Forward Translation File

```
$
$ type BRUDEN_COM.DB
$ORIGIN bruden.com.
@      IN SOA den255.bruden.com. system.bruden.com.
{
    1      ; Serial Number
           ;- Increase each time this file is changed
    600    ; Refresh time
           ;- Zone will be refreshed every 10 mins.
    300    ; Retry
           ;- Failed refresh should be retried every 5 mins.
    172800 ; Expire
           ;- Time after which zone is no longer authoritative
           ;- due to lack of refresh.
    43200} ; Minimum
           ;- minimum time to live for exported info.
;   The above five fields could also appear as ( 1 600 300 172800 43200).

alfman    IN      A      192.206.126.3
alfman-sl IN      A      192.206.127.3
den255    IN      A      192.206.128.27
den255-sl IN      A      192.206.127.27
compcom   IN      A      192.206.126.80
cuff      IN      A      192.206.126.4
          IN      MX     10    cuff.bruden.com
                                   ; Lower MX value is preferred mail server.
          IN      MX     100   alfman.bruden.com

$
```



Note

This file has no NS record. BIND produces an error in its log file. This issue is covered later in the module.

\$

\$ **dir**

Directory SYS\$SPECIFIC:[TCPIP\$BIND]

126_206_192_IN-ADDR_ARPA.DB;1	127_0_0.DB;1
127_206_192_IN-ADDR_ARPA.DB;1	128_206_192_IN-ADDR_ARPA.DB;1
BRUDEN_COM.DB;2	BRUDEN_COM.DB;1
LOGIN.COM;1	LOCALHOST.DB;1
ROOT.HINT;1	
TCPIP\$BIND.CONF;3	TCPIP\$BIND.CONF;2
TCPIP\$BIND.CONF;1	TCPIP\$BIND.CONF;1
TCPIP\$BIND_CONF.TEMPLATE;1	TCPIP\$BIND_RUN.LOG;1

Total of 14 files.

\$

Reverse Translation Files

\$ type 128_206_192_IN-ADDR_ARPA.DB

\$ORIGIN 128.206.192.in-addr.arpa.

@ IN SOA den255.bruden.com. system.bruden.com. (
1 600 300 172800 43200)

;

IN NS 127.0.0.1.
27 IN PTR den255.bruden.com.

\$ type 126_206_192_IN-ADDR_ARPA.DB

\$ORIGIN 126.206.192.in-addr.arpa.

@ IN SOA den255.bruden.com. system.bruden.com. (
1 600 300 172800 43200)

IN NS 127.0.0.1.
3 IN PTR alfman.bruden.com.
80 IN PTR compcom.bruden.com.
4 IN PTR cuff.bruden.com.

\$ type 127_206_192_IN-ADDR_ARPA.DB

\$ORIGIN 127.206.192.in-addr.arpa.

@ IN SOA den255.bruden.com. system.bruden.com. (
1 600 300 172800 43200)

IN NS 127.0.0.1.
3 IN PTR den255-sl.bruden.com.
27 IN PTR alfman-sl.bruden.com.

TCPIP> **set name /noserver=alfman**

Alfman is no longer the BIND server.

TCPIP> **set name /server=localhost**

The server is den255.

TCPIP> **sho name**

BIND Resolver Parameters

Local domain: bruden.com

System

State: Started, Enabled

Transport: UDP

Domain: bruden.com

Retry: 4

Timeout: 4

Servers: den255

(...)

Log File

TCPIP>

TCPIP> **sho host**

LOCAL database

Host address	Host name
--------------	-----------

192.206.126.3	ALFMAN, alfman
---------------	----------------

192.206.127.3	ALFMAN-SL
---------------	-----------

127.0.0.1	LOCALHOST, localhost
-----------	----------------------

192.206.128.27	den255, DEN255
----------------	----------------

%TCPIP-W-NORECORD, information not found

No BIND information!

-TCPIP-E-BIND_READERR, error reading/interpreting query response

TCPIP>

TCPIP> **exit**

\$

\$ **type TCPIP\$BIND_RUN.LOG**

Errors reported in the log.

\$ Set NoOn

\$ VERIFY = F\$VERIFY(F\$TRNLNM("SYLOGIN_VERIFY"))

Thu 24 09:53:27 NOTICE: starting. Digital TCP/IP Services for OpenVMS - BIND
8.1.2 V5.0-9

(...)

You must add missing NS records to the bruden_com.db file.

Thu 24 09:53:28 WARNING: Zone "bruden.com" (file bruden_com.db): no NS RRs found
at zone top

Thu 24 09:53:28 WARNING: master zone "bruden.com" (IN) rejected due to errors
(serial 0)

(...)

\$

Add the following after the SOA record in bruden_com.db:

IN	NS	127.0.0.1
----	----	-----------

Server Access from Client

\$ sho sys

OpenVMS V7.2 on node DEN255 24-JUN-1999 10:57:52.65 Uptime 0 03:47:46

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
(...)							

The BIND process exists but files have errors.

0000032F	TCPIP\$BIND	LEF	8	3264	0 00:00:04.50	656	254 N
00000330	TCPIP\$NTP	LEF	9	2709	0 00:00:00.33	369	197 N
(...)							

\$

After adding missing NS records to files and restarting BIND.

\$

\$ rsh alfman cat /etc/hosts

#

#192.206.126.4 cuff

The line in /etc/hosts with cuff information is commented out.

(...)

\$ rsh alfman "/sbin/ping cuff"

Found it anyway with the help of the BIND server on den255.

```
PING cuff.bruden.com (192.206.126.4): 56 data bytes
64 bytes from 192.206.126.4: icmp_seq=0 ttl=64 time=1 ms
64 bytes from 192.206.126.4: icmp_seq=1 ttl=64 time=1 ms
64 bytes from 192.206.126.4: icmp_seq=2 ttl=64 time=1 ms
64 bytes from 192.206.126.4: icmp_seq=3 ttl=64 time=1 ms
%RSH-S-LCLCLOSED, Local connection closed
```

\$

\$

\$ rsh alfman "/usr/bin/nslookup cuff"

Server: den255.bruden.com

Address: 192.206.128.27

Name: cuff.bruden.com

Address: 192.206.126.4

\$

\$

\$ rsh alfman "/sbin/grep BIND /etc/rc.config"

BIND_CONF="YES"

export BIND_CONF

BIND_SERVERTYPE="CLIENT"

Node alfman is configured as a BIND client.

export BIND_SERVERTYPE

BIND_SERVERARGS="NONE"

export BIND_SERVERARGS

\$

\$

Looking at Resource Records

```
$ nslookup:=$sys$system:tcpip$nslookup.exe
```

Create a foreign command.

```
$
```

```
$
```

```
$ nslookup
```

Default Server: den255.bruden.com

Address:

192.206.128.27

```
> ls -d bruden.com
```

[den255.bruden.com]

\$ORIGIN bruden.com.

```
@          1D IN SOA      @ system.den255 (
                                1          ; serial
                                10M        ; refresh
                                5M         ; retry
                                1w6d13h46m40s ; expiry
                                12H )       ; minimum
                                12H IN NS    den255
den255      12H IN A      192.206.128.27
alfman      12H IN A      192.206.126.3
den255-sl   12H IN A      192.206.127.27
alfman-sl   12H IN A      192.206.127.3
compcom     12H IN A      192.206.126.80
cuff        12H IN MX     10 cuff
            12H IN MX     100 alfman
            12H IN A      192.206.126.4
@          1D IN SOA      @ system.den255 (
                                1          ; serial
                                10M        ; refresh
                                5M         ; retry
                                1w6d13h46m40s ; expiry
                                12H )       ; minimum
```



```
> ls -d 128.206.192.in-addr.arpa
[den255.bruden.com]
128.206.192.in-addr.arpa. 12H IN SOA  @ system.den255 (
                                1                ; serial
                                10M               ; refresh
                                5M                ; retry
                                1w6d13h46m40s    ; expiry
                                12H )             ; minimum
                                12H IN NS      den255
4.128.206.192.in-addr.arpa. 12H IN PTR  cuff
                                Reverse addresses.
27.128.206.192.in-addr.arpa. 12H IN PTR  den255
```

```
> exit
$
```

Correcting Database File Errors

Edit the files based on errors in the log file.

```
$
$ @sys$startup:tcpip$shutdown
(...)
$
$ @sys$startup:tcpip$startup
(...)
$
```



Note

The TCPIP command SET NAME/INIT also recognizes changes to the BIND databases.

```
$
$ type bruden_com.db
$ORIGIN bruden.com.
@      1D      IN SOA bruden.com. system.den255.bruden.com.
(...)
      IN      NS      127.0.0.1
```

The above line was missing earlier.

```
;
alfman      IN      A      192.206.126.3
alfman-sl   IN      A      192.206.127.3
den255      IN      A      192.206.128.27
den255-sl   IN      A      192.206.127.27
compcom     IN      A      192.206.126.80
cuff        IN      A      192.206.126.4
            IN      MX      10      cuff.bruden.com.
            IN      MX      100     alfman.bruden.com.
$
```

Show Server Characteristics

TCPIP>

TCPIP> **sho name**

BIND Resolver Parameters

Local domain: bruden.com

System

State: Started, Enabled

Transport: UDP

Domain: bruden.com

Retry: 4

Timeout: 4

Servers: den255

Path: No values defined

Process

State: Enabled

Transport:

Domain:

Retry:

Timeout:

Servers:

Path:

TCPIP>

Accessing Server Database Files

TCPIP> sho host

LOCAL database

Host address	Host name
192.206.126.3	ALFMAN, alfman
192.206.127.3	ALFMAN-SL
127.0.0.1	LOCALHOST, localhost
192.206.128.27	den255, DEN255

BIND database

Server: 192.206.128.27 den255

Host address	Host name
192.206.128.27	den255.bruden.com
192.206.126.3	alfman.bruden.com
192.206.127.27	den255-sl.bruden.com
192.206.127.3	alfman-sl.bruden.com
192.206.126.80	compcom.bruden.com
192.206.126.4	cuff.bruden.com

TCPIP>

TCPIP>

TCPIP>

TCPIP> **sho host alfman-sl**

LOCAL database

Host address	Host name
--------------	-----------

192.206.127.3	ALFMAN-SL
---------------	-----------

This is from the local database.

TCPIP>

TCPIP>

TCPIP> **sho host alfman-sl/server=den255**

BIND database

This is from the BIND server.

Server: 192.206.128.27 den255

Host address	Host name
--------------	-----------

192.206.127.3	ALFMAN-SL.bruden.com
---------------	----------------------

TCPIP>

TCPIP>

Adding a New Node

Edit the files and add a new node.

TCPIP> **set name /init**

Rereads the database files (no new log created).

TCPIP> **sho host**

LOCAL database

Host address	Host name
192.206.126.3	ALFMAN, alfman
192.206.127.3	ALFMAN-SL
127.0.0.1	LOCALHOST, localhost
192.206.128.27	den255, DEN255

BIND database

Server: 192.206.128.27 den255

Host address	Host name
192.206.128.27	den255.bruden.com
192.206.126.3	alfman.bruden.com
192.206.127.27	den255-sl.bruden.com
192.206.127.3	alfman-sl.bruden.com
192.206.126.80	compcom.bruden.com
<u>192.206.126.19</u>	<u>penman.bruden.com</u>

New node added.

192.206.126.4 cuff.bruden.com

TCPIP>

TCP/IP> **sho host penman**

BIND database

Server: 192.206.128.27 den255

Host address Host name

192.206.126.19 PENMAN.bruden.com

TCP/IP>

TCP/IP> **sho host /addr=192.206.126.19**

Reverse lookup.

BIND database

Server: 192.206.128.27 den255

Host address Host name

192.206.126.19 penman.bruden.com

TCP/IP>

TCP/IP> **exit**

Cluster Load Balancing



Note

Cluster, in this context, means a group of name server IP addresses all referenced by the same name.

- Round-robin scheduling is the default method used by name servers.
- When multiple resource records satisfy a query, the server returns them in a round-robin order.
 - Client requests rotate to one cluster member, and then the next client request rotates to the next cluster member.
 - The cluster could share mail loads among multiple equivalent gateways.
 - This does not take into account the current workload and does not know if a host is up or down, so there is no real load balancing.

Portion of domain.db File

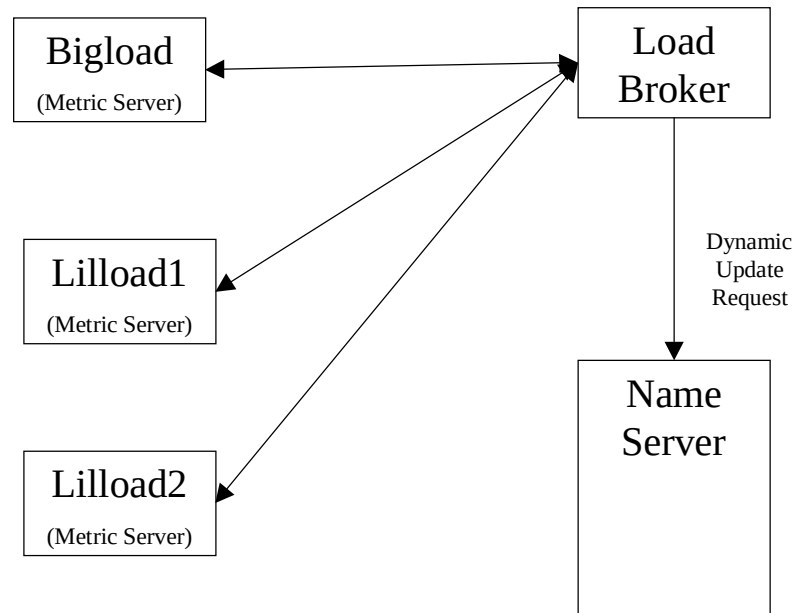
```
;
; TCP/IP cluster round robin scheduling
; cluster name: "rndrobin"
rndrobin                IN      A      192.206.126.3 ;alfman
                        IN      A      192.206.126.4 ;cuff
                        IN      A      192.206.128.27 ;den255
;
```

- Telnet *rndrobin* proceeds in the following order:
 - 192.206.126.3 (alfman)
 - 192.206.126.4 (cuff)
 - 192.206.128.27 (den255)
 - 192.206.126.3 (alfman)

Load Broker

Load broker is a configurable, calculated, load-balancing mechanism for distributing the load among cluster members.

- Load broker considers the load on all cluster members before choosing a server to handle the request.
- Load broker polls cluster members and updates the name server.
 - The name server must allow dynamic updates from the node on which the load broker is running. The configuration file would indicate *allow-updates*.
 - Each member of the cluster must be running the *metric server* code.
- The following figure shows load broker polling metric servers and dynamically updating the name server.



Dynamic Update with Load Broker

Configuring the Load Broker and Metric Server

- Configure the metric server through TCPIP\$CONFIG. Start it with:
\$ @SYS\$STARTUP:TCPIP\$METRIC_STARTUP.COM
- Configure the load broker:
 - Edit TCPIP\$LBROKER_CONF.TEMPLATE in
SYS\$SYSDEVICE:[TCPIP\$LD_BKR].
 - Rename it TCPIP\$LBROKER.CONF.
- Sample TCPIP\$LBROKER.CONF:

```
cluster "rndrobin.bruden.com"
```

Cluster name.

```
{
```

```
dns-ttl 45;
```

Server cache time.

```
dns-refresh 30;
```

Server refresh interval.

```
masters {
```

```
192.206.128.27;
```

Authoritative server.

```
};
```

```
polling-interval 30;
```

Time between poll of cluster members.

```
max-members 2;
```

Maximum number of addresses to be dynamically returned.

```
members {
```

```
192.206.128.27;
```

Cluster members.

```
192.206.126.3;
```

```
192.206.126.4;
```

```
192.206.126.34;
```

```
192.206.126.45;
```

```
};
```

```
failover 192.206.126.40;
```

Use if all others are down.

```
};
```

Load Broker and Metric Server Interaction

- Load broker polls cluster members (from the configuration file) upon startup.
- Load broker communicates with metric server software running on each cluster node.
 - Metric server repeatedly (every `TCPIP$METRIC_COMPUTE_INTERVAL` seconds - default 10) calculates system load and reports when contacted by the load broker.
 - Load is based on memory usage, CPU power, and an arbitrary administrative bias set with a logical name (`TCPIP$METRIC_CPU_RATING`).
- Load broker creates an address list based on the result of a SORT of the load information retrieved from metric servers.

If necessary, the load broker sends a dynamic update request to the name server to update the name server's list of addresses corresponding to *rndrobin.bruden.com* to reflect the load data reported by the metric servers.

Bootstrap Protocol (BOOTP)

Module 9

Introduction

This module discusses downloading software from a server to a client. This allows for diskless workstations that request a system image from a server. Client and server sides are discussed conceptually, and then configured. Several traces are used to follow the action as the download requests are fielded by the server.

Objectives

Upon completion of this module, you should be able to:

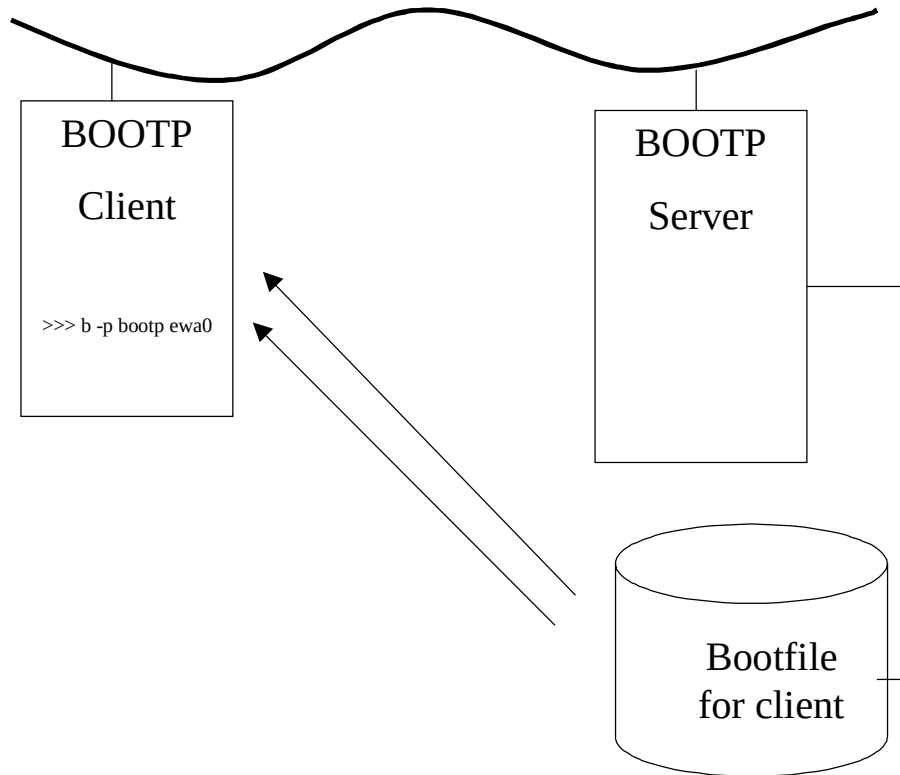
- Describe the BOOTP protocol.
- Prepare a server for download responsibilities.
- Discuss controller actions within BOOTP context.
- Describe the relationship between BOOTP and TFTP.

Topics

- BOOTP Concepts
- BOOTP Server
- BOOTP Client
- Server Configuration
- Server Data Files
- TFTP

BOOTP (Bootstrap Protocol)

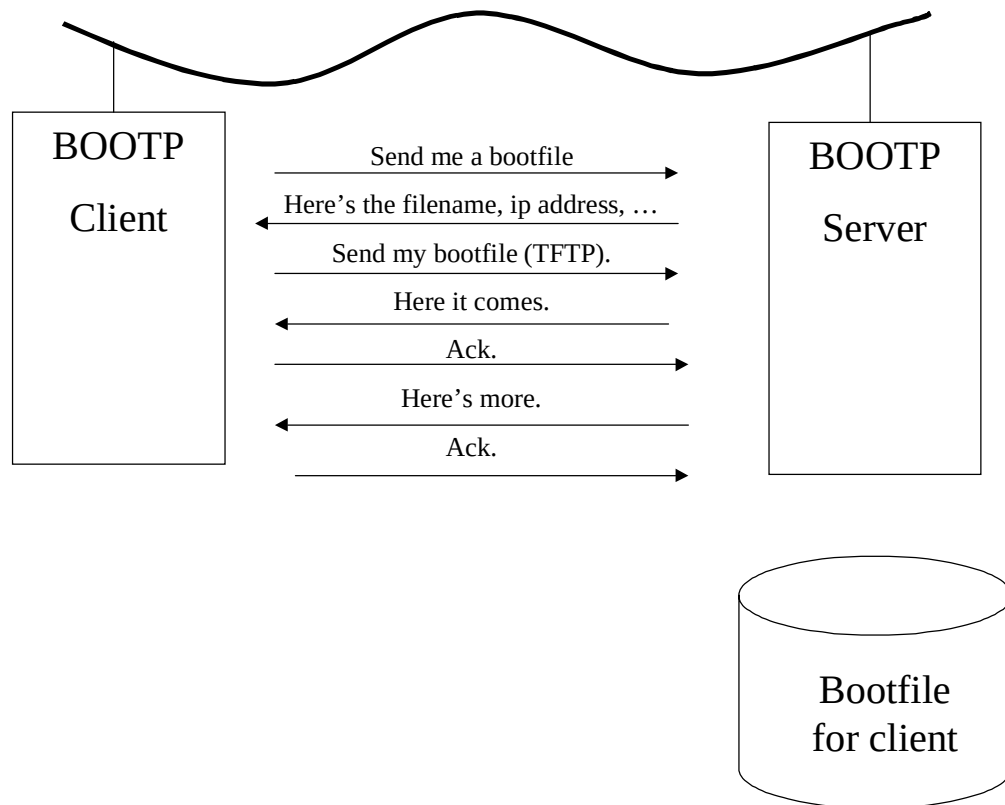
- A client machine can request that its operating system be downloaded at boot time.
 - It would boot from the network device, not the disk.
 - This is useful for diskless workstations.
- The network controller must be able to handle the BOOTP protocol. Use the following hardware-specific command:
`>>> b -p bootp ewa0`
- In the following figure, the server responds to a BOOTP request from the client.



Client BOOTP Request

BOOTP Server

- The BOOTP server does not actually use the BOOTP protocol to send down the operating system software.
- The BOOTP server responds with information describing what the client should download and other identification data.
 - Client IP address
 - Client host name (usually)
 - Name and size of the client system load file
 - IP address of the TFTP server storing this file
 - IP addresses of the hosts offering common network services



BOOTP Client and Server Initial Communications

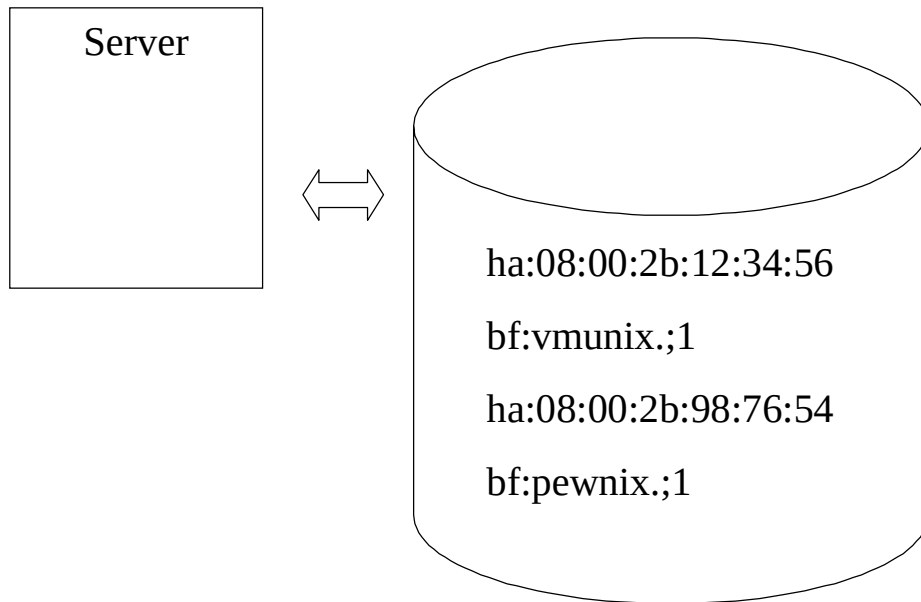
TFTP Download

- The client typically uses the TFTP protocol to do the download.
 - TFTP separates the *configuration* step from the *storage* step, allowing the administrator to configure sets of machines or individual machines.
 - The server must identify the client.
 - However, the client has no host name or IP address.
 - The client uses its MAC address to distinguish itself from other clients.

**Note**

Some systems (Windows 95, Windows NT, and Windows for Workgroups 3.11) precede the MAC address with a type byte—01 is typical.

- The server has a database keyed by the MAC address, containing specifics about each client.



Controller Activities

- Most Ethernet controllers default to MOP.
 - Can be altered to BOOTP at the console prompt (>>>).
 - Requires ROM based intelligence in the Ethernet card.
 - Must understand BOOTP, IP, UDP, and TFTP.
- Sends to 255.255.255.255, which is a special-case, limited-broadcast address.
 - IP can use this destination address before learning anything about its network environment.
 - Uses 0.0.0.0 for the source address.
 - IP network address of 0 means *this network*, whatever it might be.
 - IP node address of 0 means *this node*, whatever its number might be.
- The BOOTP client uses UDP with timers to determine the status of the request.
 - Will time out and retransmit using a random delay to avoid collisions.
 - Consider a room full of BOOTP clients after a powerfail event!

BOOTP Commands and Logicals

BOOTP Management Commands

Command	Description
CONVERT /VMS BOOTP	Populates an existing BOOTP database with entries from a UNIX /etc/bootptab file.
CREATE BOOTP	Creates an empty BOOTP database.
SET BOOTP	Adds or modifies client entries to the BOOTP database.
SHOW BOOTP	Displays information from the BOOTP database.
ENABLE SERVICE BOOTP	Interactively enables the BOOTP service.
DISABLE SERVICE BOOTP	Interactively disables the BOOTP service.
SET CONFIGURATION ENABLE SERVICE BOOTP	Sets the configuration database to enable BOOTP at startup.
SET SERVICE BOOTP	Configures the service in the services database.
SHOW SERVICE BOOTP	Displays BOOTP information stored in the services database.

BOOTP and TFTP Logical Names

Logical Name	Description
TCPIP\$BOOTP	Points to the location of the BOOTP database file.
TCPIP\$TFTP_ROOT	Defines a concealed device. Points to the TFTP data storage tree, SYS\$SYSDEVICE:[TCPIP\$TFTP_ROOT.].

BOOTP Server

Adding Entries to the Server Database

- To create individual entries:
TCPIP> SET BOOTP host /FILE=download_file/HARDWARE=ADDRESS=hex_address
- To convert entries from a UNIX /etc/bootptab file:
\$ CONVERT /VMS BOOTP

Download File Location

- By default, BOOTP looks for the download file in TCPIP\$TFTP_ROOT:[host].
- Otherwise, it looks in TCPIP\$TFTP_ROOT:[000000].

Setting Up the BOOTP and TFTP Services

TCPIP\$CONFIG creates:

- BOOTP user account.
- TFTP user account.
- Service records in the services database.
- Default directories.
- TFTP root directory, to which the logical name TCPIP\$TFTP_ROOT: points.

Enabling and Disabling BOOTP and TFTP

- On the running system:
TCPIP> ENABLE SERVICE BOOTP
TCPIP> ENABLE SERVICE TFTP
- In the configuration database:
TCPIP> SET CONFIGURATION ENABLE SERVICE BOOTP
TCPIP> SET CONFIGURATION ENABLE SERVICE TFTP
- To monitor service states:
TCPIP> SHOW SERVICE BOOTP
TCPIP> SHOW SERVICE TFTP
TCPIP> SHOW CONFIGURATION ENABLE SERVICE BOOTP
TCPIP> SHOW CONFIGURATION ENABLE SERVICE TFTP

TFTP Characteristics

- TFTP clients are not registered in a database, and they run as unprivileged users in the TCPIP\$TFTP account.
- File access is restricted to those that an unprivileged user can access.
- TFTP has little security (no user name or password).

TFTP Management Commands

Command	Description
ENABLE SERVICE TFTP	Enables the service.
DISABLE SERVICE TFTP	Disables the service.
SET SERVICE TFTP	Configures TFTP in the service database.
SHOW SERVICE TFTP	Displays information about TFTP from the service database.

BOOTP Configuration and Data File

\$

\$ **dir sys\$system:tcpip\$b***

Note, no data file in here.

Directory SYS\$COMMON:[SYSEXE]

TCPIP\$BGDRIVER.STB;1

TCPIP\$BIND_RUN.COM;1

TCPIP\$BIND_SERVER.EXE;1

TCPIP\$BIND_SERVER_CONTROL.EXE;1

TCPIP\$BIND_SERVER_XFER.EXE;1

TCPIP\$BOOTP.EXE;1

TCPIP\$BOOTP_RUN.COM;1

Total of 7 files.

\$

\$

\$ **tcpip**

No BOOTP information.

TCPIP> **sho serv bootp**

%TCPIP-W-NORECORD, information not found

-RMS-E-RNF, record not found

TCPIP>

TCPIP>

No BOOTP service.

TCPIP> **sho serv**

Service	Port	Proto	Process	Address	State
BIND	53	TCP,UDP	TCPIP\$BIND	0.0.0.0	Disabled
DHCP	67	UDP	TCPIP\$DHCP	0.0.0.0	Enabled
ESNMP	242	UDP	ESNMP	0.0.0.0	Disabled
FINGER	79	TCP	TCPIP\$FINGER	0.0.0.0	Enabled
FTP	21	TCP	TCPIP\$FTP	0.0.0.0	Enabled
LBROKER	6570	UDP	TCPIP\$LBROKER	0.0.0.0	Enabled
LPD	515	TCP	TCPIP\$LPD	0.0.0.0	Enabled
MOUNT	10	UDP	TCPIP\$NFS_M	0.0.0.0	Enabled
NFS	2049	UDP	TCPIP\$NFS	0.0.0.0	Enabled
NTP	123	UDP	TCPIP\$NTP	0.0.0.0	Enabled
POP	110	TCP	TCPIP\$POP	0.0.0.0	Enabled
PORTMAPPER	111	TCP,UDP	TCPIP\$PORTM	0.0.0.0	Enabled
REXEC	512	TCP	TCPIP\$REXEC	0.0.0.0	Enabled
RLOGIN	513	TCP	not defined	0.0.0.0	Enabled
RSH	514	TCP	TCPIP\$RSH	0.0.0.0	Enabled
SMTP	25	TCP	TCPIP\$SMTP	0.0.0.0	Enabled
SNMP	161	UDP	TCPIP\$SNMP	0.0.0.0	Enabled
TELNET	23	TCP	not defined	0.0.0.0	Enabled

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

Configuration options:

- 1 - Core environment
- 2 - Client components
- Select Server components.
- 3 - Server components
- 4 - Optional components

- 5 - Shutdown DIGITAL TCP/IP Services for OpenVMS
- 6 - Startup DIGITAL TCP/IP Services for OpenVMS
- 7 - Run tests

- A - Configure options 1 - 4
- [E] - Exit configuration procedure

Enter configuration option: **3**

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Disabled
- Select BOOTP.
- 2 - BOOTP Disabled
- 3 - TFTP Disabled
- (...)

Enter configuration option: **2**

BOOTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

BOOTP SERVER configuration options:

Select Enable service.

1 - Enable service on this node

[E] - Exit BOOTP configuration

Enter configuration option: **1**

The TCPIP\$AUX identifier already exists with value 3655

Found DHCP Service Entry. Disabling and Deleting it...

Creating BOOTP Service Entry

Data file creation.

Creating SYSS\$COMMON:[SYSEXEC]TCPIP\$BOOTP.DAT

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

1 - BIND Disabled

2 - BOOTP Enabled

3 - TFTP Disabled

(...)

Enabling TFTP

Enter configuration option: **3**

TFTP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

TFTP SERVER configuration options:

Select Enable service.

1 - Enable service on this node

[E] - Exit TFTP configuration

Enter configuration option: **1**

Creating TFTP Service Entry

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

1	-	BIND	Disabled
2	-	BOOTP	Enabled
3	-	<u>TFTP</u>	<u>Enabled</u>

(...)

\$ @sys\$startup:tcpip\$shutdown

(...)

\$ @sys\$startup:tcpip\$startup

(...)

Command Examples

\$ **tcpip**

TCPIP>

TCPIP>

Brief service display.

TCPIP> **sho serv bootp**

Service	Port	Proto	Process	Address	State
BOOTP	67	UDP	TCPIP\$BOOTP	0.0.0.0	Enabled

TCPIP>

TCPIP>

Full service display.

TCPIP> **sho serv bootp/full**

Note the BOOTP port number 67.

Service: BOOTP

State: Enabled

Port: 67 Protocol: UDP Address: 0.0.0.0

Inactivity: 5 User_name: TCPIP\$BOOTP Process: TCPIP\$BOOTP

Limit: 1 Active: 0 Peak: 0

File: TCPIP\$SYSTEM:TCPIP\$BOOTP_RUN.COM

Flags: Listen

Socket Opts: Rcheck Scheck

Receive: 0 Send: 0

Log Opts: Acpt Actv Dactv Conn Error Exit Logi Logo Mdfy Rjct Tim0 Addr

File: SYS\$SYSDEVICE:[TCPIP\$BOOTP]TCPIP\$BOOTP_RUN.LOG

Security

Reject msg: not defined

Accept from any host.

Accept host: 0.0.0.0

Accept from any network.

Accept netw: 0.0.0.0

TCPIP>

TCPIP> **sho serv tftp**

Service	Port	Proto	Process	Address	State
TFTP	<u>69</u>	UDP	TCPIP\$TFTP	0.0.0.0	Enabled

TCPIP> **sho serv tftp/full**

Note the TFTP port number 69.

Service: TFTP

State: Enabled

Port: 69 Protocol: UDP Address: 0.0.0.0

Inactivity: 5 User_name: TCPIP\$TFTP Process: TCPIP\$TFTP

Limit: 1 Active: 0 Peak: 0

File: TCPIP\$SYSTEM:TCPIP\$TFTP_RUN.COM

Flags: Listen

Socket Opts: Rcheck Scheck

Receive: 0 Send: 0

Log Opts: Acpt Actv Dactv Conn Error Exit Logi Logo Mdfy Rjct Tim0 Addr

File: SYS\$SYSDEVICE:[TCPIP\$TFTP]TCPIP\$TFTP_RUN.LOG

Security

Reject msg: not defined

Accept host: 0.0.0.0

Accept netw: 0.0.0.0

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$

TFTP File Configuration

```
$ sho log tcpip$t*
```

```
(...)
```

```
(LNM$SYSTEM_TABLE)
```

Concealed device.

```
"TCPIP$TFTP_ROOT" = "SYS$SYSDEVICE:[TCPIP$TFTP_ROOT.]"
```

```
$
```

Create a download directory for node cuff.

```
$ create/dir tcpip$tftp_root:[cuff]
```

```
$
```

Copy the system file over from cuff.

```
$ rcp /user="root" /pass="Bruden" "cuff:/vmunix" tcpip$tftp_root:[cuff]
```

```
$
```

```
$ dir tcpip$tftp_root:[cuff]/size
```

```
Directory TCPIP$TFTP_ROOT:[CUFF]
```

Size is approximately 9M.

```
VMUNIX.;1          18198
```

Total of 1 file, 18198 blocks.

```
$
```

```
$ rsh /user="root" cuff "ls -l /vmunix"
```

Size is approximately 9M.

```
-rwxr-xr-x  1 root      system  9316960 Jun  1  1998 /vmunix
```

```
$
```

```
$ tcpip
```

```
TCPIP>
```

```
TCPIP> sho serv
```

Service	Port	Proto	Process	Address	State
BOOTP	67	UDP	TCPIP\$BOOTP	0.0.0.0	Enabled
(...)					
TFTP	69	UDP	TCPIP\$TFTP	0.0.0.0	Enabled

```
TCPIP>
```

BOOTP Data File Configuration

TCPIP>

Add data to SYS\$SYSTEM:TCPIP\$BOOTP.DAT.

TCPIP>

TCPIP> **set bootp cuff /hardware=address=08-00-2b-e7-71-58 /file=vmunix.;1**

TCPIP>

TCPIP>

TCPIP>

Show data from TCPIP\$BOOTP.DAT

TCPIP> **show bootp/full**

Host: 192.206.126.4 CUFF

MAC ID.

Hardware Address: 08-00-2B-E7-71-58

Network mask: 255.255.255.0

Type: Ethernet

Download file name.

File: VMUNIX.;1

Time offset: 0

Vendor:

Gateways: not defined

Servers:

Cookie: not defined

IEN: not defined

Impress: not defined

Log: not defined

LPR: not defined

Name: not defined

Resource: not defined

Time: not defined

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$

Trace During Ethernet Boot of Cuff (Tru64 UNIX V4.0e)

Enter the following command at boot prompt on cuff:

```
>>> b -p bootp ewa0
```

\$ **tcptrace**

The first packet is from (0.0.0.0) this network, this node.

Destination is (255.255.255.255) any node that is listening.

(...)

TCPIPtrace RCV packet 8 at 30-JUN-1999 20:39:38.50

	IP Address	Port
	Note the ports and addresses.	
Source	<u>0.0.0.0</u>	<u>68</u>
	Port 67 is the BOOTP server.	
	Port 68 is the BOOTP client.	
Destination	<u>255.255.255.255</u>	<u>67</u>

	Packet Length
UDP	328

	Hex	Count	Ascii
-----	-----	----	-----
00000000 B3041120 0000F394 48010045 0000			E..H....
00060101 8E3A3401 43004400 FFFFFFFF 0010			D.C.4:.....
00000000 00000000 00000000 77DAB93B 0020			;..W.....
00005871 E72B0008 00000000 00000000 0030			+.qX..
00000000 00000000 00000000 00000000 0040			

(...)

TCPIPtrace XMT packet 9 at 30-JUN-1999 20:39:39.24

IP Address Port

OpenVMS BOOTP server (den255).

Source 192.206.126.27 67

Note the real destination address.

Destination 192.206.126.4 68

Packet Length 328

UDP

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
1B7ECEC0	1918111E	0040CFC5	48010045	0000	E..H..@.....~.
00060102	DE593401	44004300	047ECEC0	0010	..~..C.D.4Y.....
047ECEC0	00000000	00000000	77DAB93B	0020	;..W.....~.
00005871	E72B0008	1B7ECEC0	1B7ECEC0	0030	..~...~...+.qX..

Sender.

622E3535	326E6564	00000000	00000000	0040	den255.b
00000000	0000006D	6F632E6E	65647572	0050	ruden.com.....
00000000	00000000	00000000	00000000	0060	
00000000	00000000	00000000	00000000	0070	

Boot file name.

46542450	49504354	00000000	00000000	0080	TCPIP\$TF
4D565D46	4655435B	3A544F4F	525F5054	0090	TP_ROOT:[CUFF]VM
00000000	00000000	0000002E	58494E55	00A0	UNIX.....
00000000	00000000	00000000	00000000	00B0	
00000000	00000000	00000000	00000000	00C0	
00000000	00000000	00000000	00000000	00D0	
00000000	00000000	00000000	00000000	00E0	
00000000	00000000	00000000	00000000	00F0	
FFFF0401	63538263	00000000	00000000	0100	c.Sc....

Receiver.

000000FF	46465543	040C1647	020D00FF	0110	G...CUFF
00000000	00000000	00000000	00000000	0120	
00000000	00000000	00000000	00000000	0130	
		00000000	00000000	0140	

Trace During Ethernet Boot of Cuff with TFTP

TCPIPtrace RCV packet 10 at 30-JUN-1999 20:39:39.37

	IP Address	Port
Source	192.206.126.4	36646
TFTP download request. Note port 69.		
Destination	192.206.126.27	<u>69</u>

Packet Length 66
UDP

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
047ECEC0	B2BC1120	00003C60	42000045	0000	E..B`<.. ..~.
43540100	A0E02E00	4500268F	1B7ECEC0	0010	..~..&.E.....TC
435B3A54	4F4F525F	50544654	24504950	0020	PIP\$TFTP_ROOT:[C
6574636F	002E5849	4E554D56	5D464655	0030	UFF] <u>VMUNIX</u> ..octe

Note the file name.

0074	0040	t.
------	------	----

\$ **tcptrace**

TCPIPtrace XMT packet 2 at 30-JUN-1999 20:39:57.74

	IP Address	Port
	From den255 (BOOTP server).	
Source	192.206.126.27	1045
	To cuff (BOOTP client).	
Destination	192.206.126.4	36646

Packet Length	544
---------------	-----

UDP

Sample file contents.

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
1B7ECEC0	D5F2111E	00403AEA	20020045	0000	E.. .:@.....~.
67240300	5E860C02	268F1504	047ECEC0	0010	..~....&...^..\$g
62A56012	1565A061	05141565	06121564	0020	d...e...a.e...`b
05141563	04121562	03133402	141565A0	0030	.e...4..b...c...
62A56012	1565A061	05141565	06121564	0040	d...e...a.e...`b
(...)					

Dynamic Host Configuration Protocol (DHCP)

Module 10

Introduction

This module adds more functions to those presented in the previous module. Dynamic Host Configuration Protocol (DHCP) is similar to BOOTP. They both use the same protocol format, but a few fields are used differently. This module describes the tools and commands available for configuring DHCP to share a pool of IP addresses among a large number of transient machines.

Objectives

Upon completion of this module, you should be able to:

- Describe the relationship between BOOTP and DHCP.
- Configure the DHCP server.
- Use the DHCP GUI.
- Describe the DHCP database files.
- Set up DHCP cluster failover.

Topics

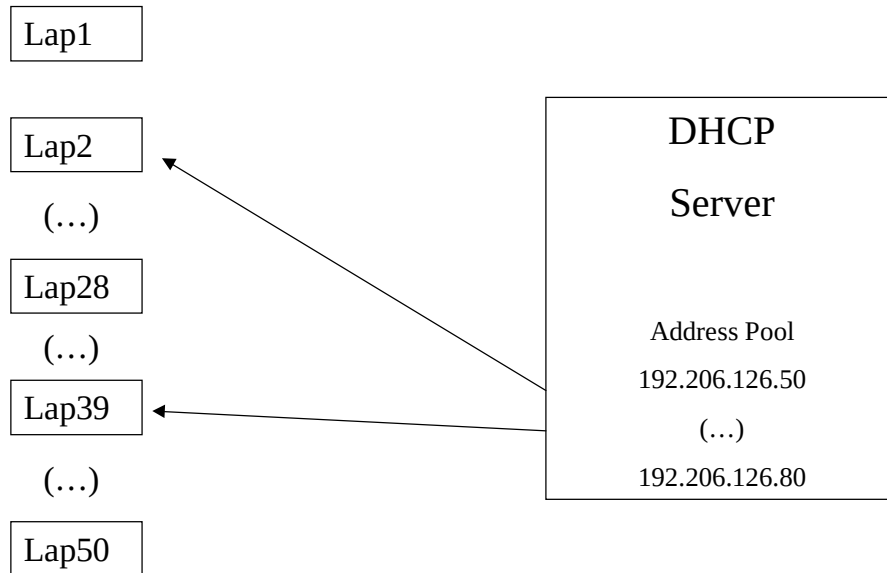
- DHCP Concepts
- Advantages of Using DHCP
- DHCP Configuration Files
- Using the DHCP GUI
- DHCP Cluster Failover

Comparing DHCP and BOOTP

- **BOOTP** provides a configuration file that defines an IP address for each client node.
 - Consider a situation where you have fewer IP addresses than nodes.
 - An example might be a college lab where you have 30 addresses to handle 50 students, each of whom has a laptop.
 - The scenario mandates that:
 - ◆ No more than 30 students can use the network at the same time.
 - ◆ IP addresses must be dynamically assigned for a finite time period.
- **DHCP** allows a computer to receive all necessary network configuration data in one message from a server node.
 - The server must be configured with a set (pool) of IP addresses.
 - The client requests one copy of the set each time it starts its network.
 - DHCP uses the same message format as BOOTP with different contents in some of the fields.
- DHCP allows static address assignment (like BOOTP), and dynamic permanent address assignment where the server permanently assigns an address when the client first starts its network.
- You can configure a DHCP server to handle all three types of assignment:
 - Static
 - Dynamic permanent
 - Dynamic leased addresses

Dynamic Address Assignment

- The administrator must prepare the set of addresses, identifying the potential clients by MAC address (like BOOTP).
- A typical lease duration is one week (configurable).



DHCP Server Leasing Addresses from Its Address Pool

Relay Agent

- Used by both BOOTP and DHCP to access a server in a remote network, the relay agent:
 - Senses the broadcast from the client and *relays* it to the appropriate server.
 - Relays the server's replies to the client.

DHCP Advantages

DHCP has the following advantages over BOOTP:

- Centralizes address administration.
- Automates IP address assignment and administration.
- Eliminates inadvertent use of duplicate addresses.
- Supports DHCP failover in an OpenVMS cluster.

DHCP Configuration

\$

\$ @sys\$startup:tcpip\$config

TCP/IP Network Configuration Procedure

(...)

Configuration options:

1 - Core environment

2 - Client components

Select Server components.

3 - Server components

4 - Optional components

5 - Shutdown DIGITAL TCP/IP Services for OpenVMS

6 - Startup DIGITAL TCP/IP Services for OpenVMS

7 - Run tests

A - Configure options 1 - 4

[E] - Exit configuration procedure

Enter configuration option: **3**

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

1 - BIND Enabled

2 - BOOTP Disabled

3 - TFTP Disabled

(...)

Not enabled yet.

17 - DHCP Disabled

A - Configure options 1 - 17

[E] - Exit menu

Enter configuration option: **17**

DHCP Configuration Files

DHCP SERVER Configuration

Service is not defined in the SYSUAF.

Service is not defined in the TCPIP\$SERVICE database.

Service is not enabled

DHCP SERVER configuration options:

Select Enable service.

1 - Enable service on this node

[E] - Exit DHCP configuration

Enter configuration option: 1

The TCPIP\$AUX identifier already exists with value 3655

Creating DHCP Service Entry

DHCP files.

Checking for DHCP template files in SYS\$SYSDEVICE:[TCPIP\$DHCP]

Describes clients.

Creating template SYS\$SYSDEVICE:[TCPIP\$DHCP]DHPCAP.

Mnemonics for DHPCAP file.

Creating template SYS\$SYSDEVICE:[TCPIP\$DHCP]DHCPTAGS.

For dynamic node names.

Creating template SYS\$SYSDEVICE:[TCPIP\$DHCP]NAMEPOOL.

For nondefault masks.

Creating template SYSSYSDEVICE:[TCPIP\$DHCP]NETMASKS.

Describes address ranges.

Creating template SYSSYSDEVICE:[TCPIP\$DHCP]NETS.

Server defaults.

Creating template SYSSYSDEVICE:[TCPIP\$DHCP]SERVER.PCY

No DHCP database files exist. This procedure can extract your old-style BOOTP entries from UCX\$BOOTP.DAT or TCPIP\$BOOTP.DAT and store them into the DHCP databases so that DHCP can serve these BOOTP clients. Only if you answer "YES" to the following question will DHCP be able to serve the BOOTP clients from your existing old-style BOOTP entries.

Do you want to extract BOOTP entries to your DHCP databases [Y]: n

Not extracting BOOTP entries to DHCP database

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

- 1 - BIND Enabled
- 2 - BOOTP Disabled
- 3 - TFTP Disabled
- (...)
- 17 - DHCP Enabled
- A - Configure options 1 - 17
- [E] - Exit menu

Enter configuration option:

(...)

```
$
$ set def sys$sysdevice:[tcpip$dhcp]
$ dir
```

Directory SYS\$SYSDEVICE:[TCP/IP\$DHCP]

```
DBA.HSH;1          DBB.BTR;1          DBC.HSH;1          DBE.BTR;1
DBF.BTR;1          DBG.HSH;1          DBH.HSH;1          DBI.HSH;1
DBJ.HSH;1          DHCPCAP.;8          DHCPTAGS.;1        JOIND.PID;1
LOGIN.COM;1        NAMEPOOL.;8        NETMASKS.;1        NETS.;8
SERVER.PCY;8       SHOW_BOOTP_FULL.OUT;1
TCP/IP$DHCP_RUN.LOG;3      TCP/IP$DHCP_RUN.LOG;2
TCP/IP$DHCP_RUN.LOG;1
```

Total of 21 files.

```
$
$
$ type DHCPCAP.;8
```

alfman:\

 ::\

 Note the cryptic mnemonics!

```
  :ht=ether:\
  :ha=08002b3980cb:\
  :ct=Alpha:\
  :ip=192.206.126.3:\
  :lt=1800:\
  :t1=600:\
  :t2=300:\
  :sm=255.255.255.0:
```

```
$
$
```

DHCPTAGS

This file contains descriptions of the mnemonics.

\$type DHCPTAGS.;1

DHCPTAGS

#

This file defines the following:

Types, descriptions and mnemonics for DHCP/BOOTP options

The display arrangement of data items within the JOIN GUI

These are the so-called "pseudo" tags and types used internally by

the JOIN server. The only useful edit you can perform on these is

to change the description. If you delete any of these or change

the tag number you will undoubtedly disable the server.

513	yi	-	pseudo	IP	-	"yiaddr"
514	sa	BOOTP	pseudo	IP	BSOG	"Boot File Server Address"
516	ht	KEY	pseudo	String	B0	"Hardware Type"
515	ha	KEY	pseudo	String	B0	"Hardware Address/Client ID"
517	nw	KEY	pseudo	IP	BS	"Net or Subnet IP Address"
519	tc	NAME/ID	pseudo	String	BSOG	"Member of Group"
520	hd	BOOTP	pseudo	String	BSOG	"Home Directory"
521	td	BOOTP	pseudo	String	BSOG	"TFTP root directory"
522	hn	BOOTP	pseudo	Boolean	BSOG	"Send client's hostname"
523	vm	BOOTP	pseudo	String	SOG	"Vendor Magic Cookie"
524	ci	-	pseudo	IP	-	"ciaddr"
525	gi	-	pseudo	IP	-	"giaddr"
526	ra	BOOTP	pseudo	IP	-	"Reply Address Override"

Standard DHCP options. These are defined by the IANA and are universally

applicable. Do **not** change the types of any of these. You may add to

the list as new options are defined. You may also change the long name.

If you wish you may change the short name but beware: the public domain

CMU bootpd requires **known** tags to operate properly.

1	sm	BOOTP	- IP	BSOG	"Subnet Mask"
2	to	BOOTP	- Time	SOG	"Time Offset"
3	gw	BOOTP	- IPList	BSOG	"Routers"
4	ts	BOOTP	- IPList	SOG	"Time Servers"
5	ns	BOOTP	- IPList	SOG	"IEN-116 Name Servers"
6	ds	BOOTP	- IPList	BSOG	"DNS Servers"
7	lg	BOOTP	- IPList	SOG	"Log Servers"
8	cs	BOOTP	- IPList	SOG	"Cookie Servers"
9	lp	BOOTP	- IPList	SOG	"LPR Servers"
10	im	BOOTP	- IPList	SOG	"Impress Servers"
11	rl	BOOTP	- IPList	SOG	"Resource Location Servers"
12	ho	BOOTP	- String	SOG	"Host Name"
13	bs	BOOTP	- Int2	BSOG	"Bootfile Size"
14	df	BOOTP	- String	SOG	"Merit Dump File"
15	dn	BOOTP	- String	BSOG	"DNS Domain Name"
16	sw	BOOTP	- IP	SOG	"Swap Server"
17	rp	BOOTP	- String	SOG	"Root Path"
18	ef	BOOTP	- String	SOG	"Extensions Path"
19	br	IP	- Boolean	SOG	"IP Forwarding"
	(...)				

NAMEPOOL, NETMASKS, and NETS

No distributed node names.

```
$ type NAMEPOOL.;10
```

```
$
```

```
$
```

```
$
```

For nondefault subnet masks.

```
$ type NETMASKS.;1
```

```
# Network masks. This file is only needed on those platforms
```

```
# which don't provide a netmasks database, either as a text
```

```
# file or as a map (NIS, NIS+, .. whatever).
```

```
#
```

```
# This file should contain an entry for each network for which
```

```
# the netmasks is other than the standard A,B or C mask. Each
```

```
# entry has two fields: the network and the mask. The network
```

```
# must be written with trailing zeros: e.g For net 192.1.1
```

```
# you do not enter
```

```
#
```

```
# 192.1.1
```

```
#
```

```
# but
```

```
#
```

```
# 192.1.1.0
```

```
#
```

```
#
```

```
# This file also supports variable subnetting: i.e. if each
# subnetted net can in turn be subnetted with a variable
# mask then the subnets can also appear on the LHS. Thus
#
#      192.1.1.0      255.255.255.224
#      192.1.1.96    255.255.255.240
#
# Network    netmask
$
$
$ type nets.
      Address range.
192.206.126.0      192.206.126.27      192.206.126.8-192.206.126.10
$
$
```

SERVER.PCY

Sets up defaults for various parameters.

```
$ type SERVER.PCY;10
```

```
# server.pcy: server side policy file.
```

```
#
```

```
# $Id: server.pcy,v 1.25 1997/02/24 06:22:45 robs Exp $
```

```
#
```

```
# DESCRIPTION
```

```
# This is a template server.pcy file. A particular site may need to make
# modifications to this, especially to the name service and name allocation
# policies in force
```

```
#
```

```
# Default time-to-live for an address lease if not specified on a
# per host, per subnet or per class basis.
```

```
default_ttl 86400
```

```
# Time to live on provisional list
```

```
provisional_ttl 60
```

```
# Size of the internal array specifying the number of address
# blocks held on the free list. This number should not be too
# high, or the server will "forget" about all previous allocations
# of expired leases very quickly. It should not be too low or
# performance will suffer.
```

```
free_list_size 8
```

```
# Define type of name service. The name service is one of
```

```
# { dns, local, nis, nis+}.
```

```
# local means use text files on the local system (i.e. /etc/hosts).
```

```
# On OpenVMS leave this option as "dns".
```

```
(...)
```

TCPIP\$DHCP_SETUPCOMMANDS.COM

\$ @sys\$startup:tcpip\$shutdown

(...)

\$ @sys\$startup:tcpip\$startup

(...)

\$

\$ show log *dhcp*

(...)

For interprocess communication.

"TCPIP\$DHCP_sig_mbx" = "MBA83:"

\$

\$ sho sys

OpenVMS V7.2 on node DEN255 29-JUN-1999 18:56:13.82 Uptime 0 10:38:44

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
00000A1C	<u>TCPIP\$DHCP</u>	LEF	8	437	0 00:00:00.81	616	252 N

(...)

\$

\$ type sys\$manager:tcpip\$dhcp_setupcommands.com

\$!

\$! File name: TCPIP\$DHCP_SETUPCOMMANDS.COM

\$! Product: Digital TCP/IP Services for OpenVMS (TCPIP)

\$! Version: V5.0

\$! Original Author: Steve Gonczi (Process)

\$! Modification History:

(...)

\$! Facility:

\$! DHCP - Dynamic Host Configuration Protocol

\$!

\$! Abstract:

\$! This command procedure defines symbols to invoke each of the DHCP utilities. For each utility from CA there is also a symbol that matches the Unix image name so that Unix system managers familiar with the CA product on Unix and its image names will feel more at home.

\$!

\$! Creation Date: 17-NOV-1997

\$!—

Two GUI commands.

\$!

\$! The DHCP GUI management utility.

\$ dhcp*gui := \$sys\$system:tcpip\$dhcp_gui.exe

\$ xjoin := \$sys\$system:tcpip\$dhcp_gui.exe

\$!

\$! Utility to send unix-style signals to the DHCP server.

\$ dhcpsig*nal := \$sys\$system:tcpip\$dhcp_signal.exe

Causes configuration files to be reread.

\$ dhcpsighup := 'dhcpsig 1

Shuts down DHCP.

\$ dhcpsigterm := 'dhcpsig 15

Dumps contents of configuration files.

\$ dhcpsigusr1 := 'dhcpsig 30

\$!

\$! Miscellaneous utilities.

Non-GUI utilities to access the DHCP databases.

\$!

\$! Dump DHCP lease DB

```
$ dhcpdbd*ump          := $sys$system:tcpip$dhcp_dbdump.exe
$ jdbd*ump             := $sys$system:tcpip$dhcp_dbdump.exe
$ dhcpdbreg*ister      := $sys$system:tcpip$dhcp_dbregister.exe
$ jdbr*eg              := $sys$system:tcpip$dhcp_dbregister.exe
$!
$! Read input from input text file and populate DHCP lease DB
$ dhcpdbmod*ify        := $sys$system:tcpip$dhcp_dbmodify.exe
$ jdbm*od              := $sys$system:tcpip$dhcp_dbmodify.exe
$ dhcpdbsh*ow          := $sys$system:tcpip$dhcp_dbshow.exe
$ jdbs*how             := $sys$system:tcpip$dhcp_dbshow.exe
$ dhcpshowdbs          := $sys$system:tcpip$dhcp_showdbs.exe
$ showdbs              := $sys$system:tcpip$dhcp_showdbs.exe
$ bpisamtoascii        := $sys$system:tcpip$dhcp_bpisamtoascii.exe
$ bpassciitodbmod      := $sys$system:tcpip$dhcp_bpassciitodbmod.exe
$ bptojdb              := $sys$system:tcpip$dhcp_bpassciitodbmod.exe
$!
$! 2.1.5 Read info from existing TCPIP$BOOTP.DAT into DHCP lease DB
$ bootptodhcp          := @sys$manager:tcpip$dhcp_bootptodhcp.com
$!
$
$
$ dir sys$sysdevice:[tcpip$dhcp]
```

Major database files are underlined.

Directory SYS\$SYSDEVICE:[TCPIP\$DHCP]

DBA.HSH;1	DBB.BTR;1	DBC.HSH;1	DBE.BTR;1
DBF.BTR;1	DBG.HSH;1	DBH.HSH;1	DBI.HSH;1
DBJ.HSH;1	<u>DHCPCAP.</u> ;3	DHCPTAGS.;1	JOIND.PID;1
LOGIN.COM;1	<u>NAMEPOOL.</u> ;3	<u>NETMASKS.</u> ;1	<u>NETS.</u> ;3
RWLOCKDBA.;1	RWLOCKDBB.;1	RWLOCKDBC.;1	RWLOCKDBF.;1
RWLOCKDBG.;1	RWLOCKDBH.;1	RWLOCKDBI.;1	RWLOCKDBJ.;1
<u>SERVER.PCY</u> ;3	TCPIP\$DHCP_RUN.LOG;1		

Total of 26 files.

\$

Process Details

\$ show proc/id=a1c

29-JUN-1999 19:29:57.87	User: <u>TCPIP\$DHCP</u>	Process ID: 00000A1C
	Node: DEN255	Process name: " <u>TCPIP\$DHCP</u> "

Terminal:

User Identifier: [TCPIP\$AUX,TCPIP\$DHCP]

Base priority: 8

Default file spec: Not available

Number of Kthreads: 1

Devices allocated: BG117:

BG118:

BG139:

BG140:

\$

Support Commands

\$

\$ @sys\$manager:tcpip\$dhcp_shutdown

\$

\$

\$ @sys\$manager:tcpip\$service_setup dhcp

\$

\$

\$

\$ tcpip enable service dhcp

BOOTP Interactions

\$

\$ **dir/full sys\$system:tcpip\$bootp.dat**

Directory SYS\$COMMON:[SYSEXE]

TCPIP\$BOOTP.DAT;1 File ID: (7016,2,0)

Size: 56/56 Owner: [1,1]

Created: 30-JUN-1999 20:03:33.13

Revised: 30-JUN-1999 20:37:27.60 (2)

Expires: <None specified>

Backup: <No backup recorded>

Effective: <None specified>

Recording: <None specified>

Indexed file.

File organization: Indexed, Prolog: 3, Using 2 keys

In 3 areas

Shelved state: Online

Caching attribute: Writethrough

File attributes: Allocation: 56, Extend: 10, Maximum bucket size: 5

Global buffer count: 0, No version limit

Contiguous best try

Record format: Variable length, maximum 2048 bytes, longest 0 bytes

Record attributes: Carriage return carriage control

RMS attributes: None

Journaling enabled: None

File protection: System:RWED, Owner:RWED, Group:RE, World:RE

Access Cntrl List: None

Client attributes: None

Total of 1 file, 56/56 blocks.

\$

\$

\$ **type sys\$system:tcpip\$bootp.dat**

Vaguely readable.

192.206.126.4 08002BE77158 (...) VMUNIX.;1

(...)

\$

\$

Changes UCX style BOOTP information into DHCP files.

\$ **@tcpip\$dhcp_bootptodhcp.com**

\$

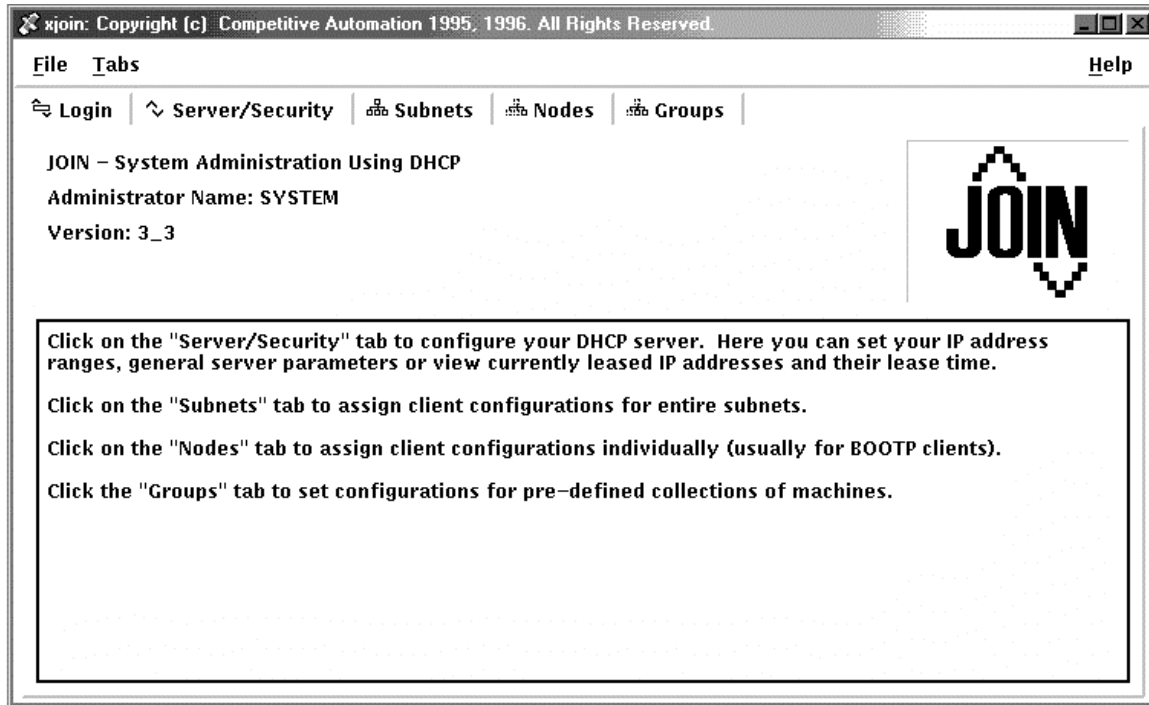
\$

\$

GUI Examples

Either command starts the GUI:

- `dhcpgui`
- `xjoin`



Parameters

The following table shows categories of parameters within the GUI.

Parameter	Description
Server/Security	Defines the server configuration. Can set IP address ranges and general server parameters, or view currently leased IP addresses and their lease time.
Subnets	Assigns client configurations for entire subnets.
Nodes	Adds and customizes specific BOOTP clients.
Groups	Defines group settings for collections of machines.

Dynamic Addressing

1. Click the *Server/Security* tab.
2. Choose *IP Ranges*, then choose *[New IP Range]*.
3. Enter the subnet address.
4. Enter the DHCP server address.
5. Enter the range of IP addresses.

The screenshot shows a window titled "dhcpd" with a menu bar (File, Tools) and a toolbar (Login, Server/Security, Subnets, Nodes, Groups). The "Server/Security" tab is selected, and the "IP Ranges" sub-tab is active. The interface is divided into three main sections:

- DHCP Server:** A list box containing "Server".
- IP Ranges:** A list box containing "[New IP Range]" and "IP Ranges for 192.206.126.0 / 192.206.126.0".
- Value(s):** A form area with the following fields:
 - Subnet Address:** 192.206.126.0
 - DHCP Server:** 192.206.126.27
 - IP Ranges:** Two input fields, the first containing "192.206.126.8" and the second containing "192.206.126.11", with "to" labels between them.

At the bottom of the window, there is a label "Enter IP Pool information" and a "Delete" button.

Static Addressing

1. Click the *Nodes* tab.
2. Choose *[New Record]*.
3. Enter the host name (*Name*).
4. Enter the MAC/hardware address (for example, 08:00:20:3f:12:4b).
5. Choose *Hardware Type* from *Key Parameters*. Enter the type of network on which the node resides (typically 1—see below).
6. Choose *[Host IP Address]*.
7. Enter the Host IP address of the host computer for this node.

Hardware Types

Symbol	Number	Network Type
ethernet1	0	Mb Ethernet
ethernet3	2	3 Mb experimental
ax.25	3	AX.25 Amateur Radio
protnet	4	Proteon ProNet Token Ring
chaos	5	Chaos
token-ring	6	IEEE802
arcnet	7	ARCNET

Static and Permanent Address Assignment

1. Click the *Server/Security* tab.
2. Choose *Active IP Snapshot*, then choose *[New Record]*.
3. Enter the MAC address.
4. Enter the MAC type.
5. Enter the MAC address length.
6. Enter an IP address that does not belong to any IP address pools.
7. Enter -1 (infinite lease) for the lease expiration.
8. Enter the Server IP address.
9. If you want a name associated with the client, specify the client's host name and domain name.

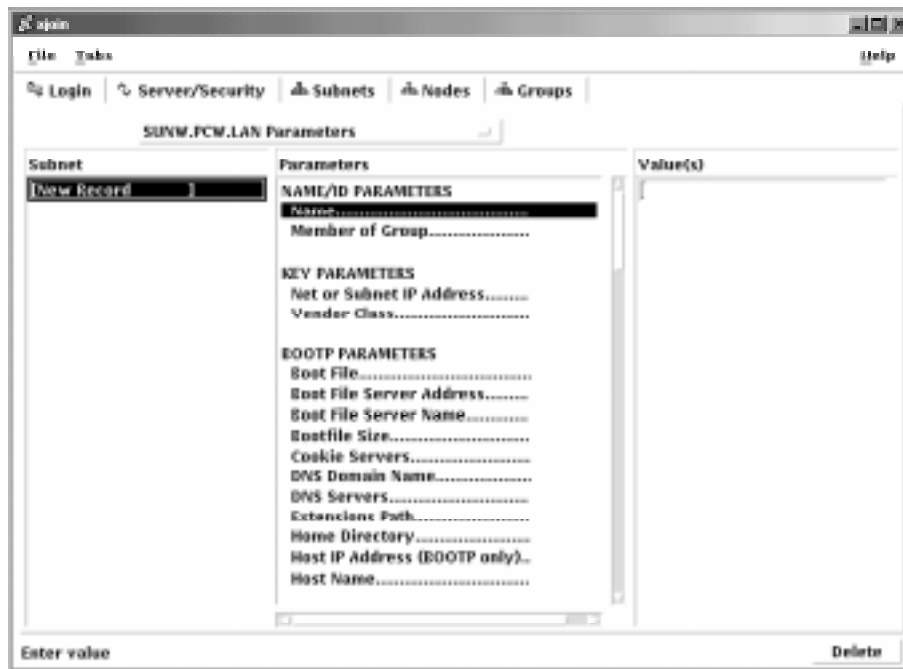


Note

Windows NT, Windows 95, and Windows for Workgroups clients precede their MAC addresses with the hardware type (such as 01:08:00:2b:12:34:56).

Configuring DHCP Parameters

1. Click the *Subnet* tab (for static clients, click the *Node* tab).
2. Select *[New Record]*.
3. Enter the subnet name (*Name*).
4. Enter the Subnet IP address.
5. Define any other DHCP parameters. By default, only basic DHCP parameters are listed. For additional parameters, click *Basic DHCP Parameters* to display the DHCP parameters you can define.



DHCP Cluster Failover Support

- Only one DHCP server process can be active within the cluster.
 - It holds an exclusive lock on the TCPIP\$DHCP_SERVER resource name.
 - Other potential servers within the cluster must wait to acquire the lock.
 - If they ever do acquire the lock, it indicates that the original server is dead and that they are the new server.
- To configure the DHCP server failover environment:
 1. Create a cluster-accessible directory for the DHCP files. Specify TCPIP\$DHCP as the directory's owner.
`$ CREATE/DIRECTORY/OWNER=TCPIP$DHCP WORK1$: [DHCP_CONFIG]`
 2. On each cluster node that might run the server code, set up the TCPIP\$DHCP_CONFIG logical name.
`$ DEFINE/SYSTEM TCPIP$DHCP_CONFIG WORK1$: [DHCP_CONFIG]`

Note

Add the TCPIP\$DHCP_CONFIG logical name definition to the SYSTARTUP_VMS.COM file before running the TCPIP\$STARTUP.COM procedure.

3. Disable DHCP.
`$ dhcpsigterm`
4. Set the auto_sync_dbs parameter in the SERVER.PCY file.
 - EDIT SERVER.PCY or use the GUI *Server/Security* option.
 - Flushes the DHCP server databases after each update.
5. Copy the DHCP data files from SYS\$SYSDEVICE:[TCPIP\$DHCP]*.* to TCPIP\$DHCP_CONFIG:*.*.
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]DHCPCAP. TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]DHCPTAGS. TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]NAMEPOOL. TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]NETMASKS. TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]NETS. TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]SERVER.PCY TCPIP$DHCP_CONFIG:`
`$ COPY SYS$SYSDEVICE:[TCPIP$DHCP]DB%.%% TCPIP$DHCP_CONFIG:`

6. Set the file ownership to TCPIP\$DHCP and protection to (O:RWED).
\$ DIRECTORY/SECURITY WORK1\$: [DHCP_CONFIG]
\$ DIRECTORY/SECURITY WORK1\$: [000000]DHCP_CONFIG.DIR
7. Remove the DHCP data files from the
SYS\$SYSDEVICE:[TCPIP\$DHCP] directory (save copies).

```
$ CREATE/DIR SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ PURGE SYS$SYSDEVICE:[TCPIP$DHCP]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]DHCCAP.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]DHCPTAGS.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]NAMEPOOL.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]NETMASKS.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]NETS.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]SERVER.PCY.* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

```
$ RENAME SYS$SYSDEVICE:[TCPIP$DHCP]DB%.%;* SYS$SYSDEVICE:[TCPIP$DHCP.SAVE]
```

8. Enable DHCP on the other cluster nodes (TCPIP\$CONFIG).
9. Edit the NETS. file and set the ownership of any existing IP address range to 0.0.0.0.
 - So no particular server owns the address range:
17.18.0.0 17.18.208.100 17.18.208.10-17.18.208.50
 - You would change it to:
17.18.0.0 0.0.0.0 17.18.208.10-17.18.208.50
 - Or, use the GUI *Server/Security* tab to set the DHCP server field to 0.0.0.0.
10. Reboot the cluster.

Introduction

This module describes how to synchronize the system times on multiple nodes on the network. This capability is critical for NFS activities and anything else that requires timing. This module presents NTP concepts and shows NTP implementation through daemon processes and support files. NTP configuration is discussed and a client example is shown, including a trace of the communication between the NTP client and server nodes.

Objectives

Upon completion of this module, you should be able to:

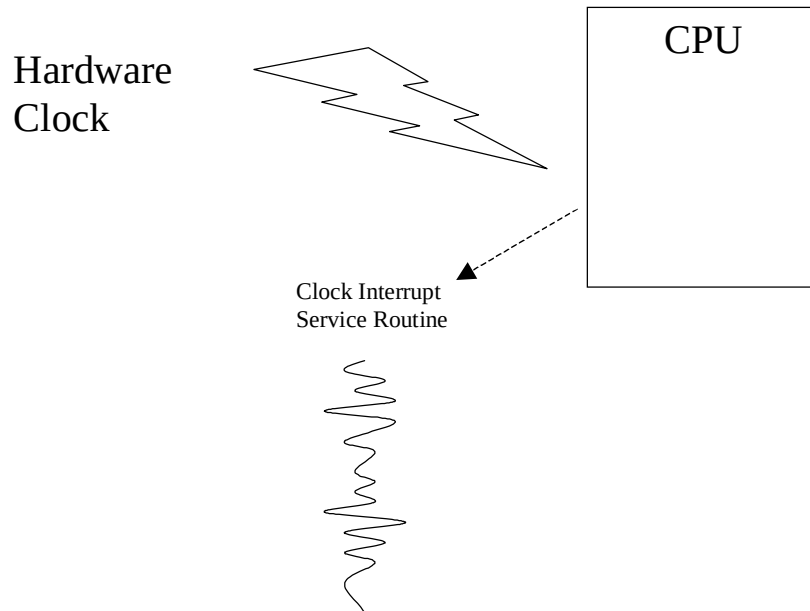
- Describe the NTP protocol.
- Discuss NTP concepts.
- Prepare the NTP configuration file.
- Use commands to manage NTP.

Topics

- NTP Concepts
- NTP Internet Time Sources
- NTP Configuration Options
- NTP Support Files
- NTP Commands
- Trace of NTP Communication

Time Synchronization

- Every computer has a means of recording the passage of time.
 - Typically this involves a repetitive interrupt of the CPU occurring many times per second (usually 1024/sec on Alpha processors).
 - The hardware clock interrupt is serviced by a software interrupt service routine (ISR).
 - The ISR is responsible for updating a kernel cell holding a binary representation of the current time.
- Due to various system events and hardware issues, the system time may *drift* from what it should be.
- In a network, each system's view of the current time can be different, and each system may drift so that its time is either too fast or too slow.

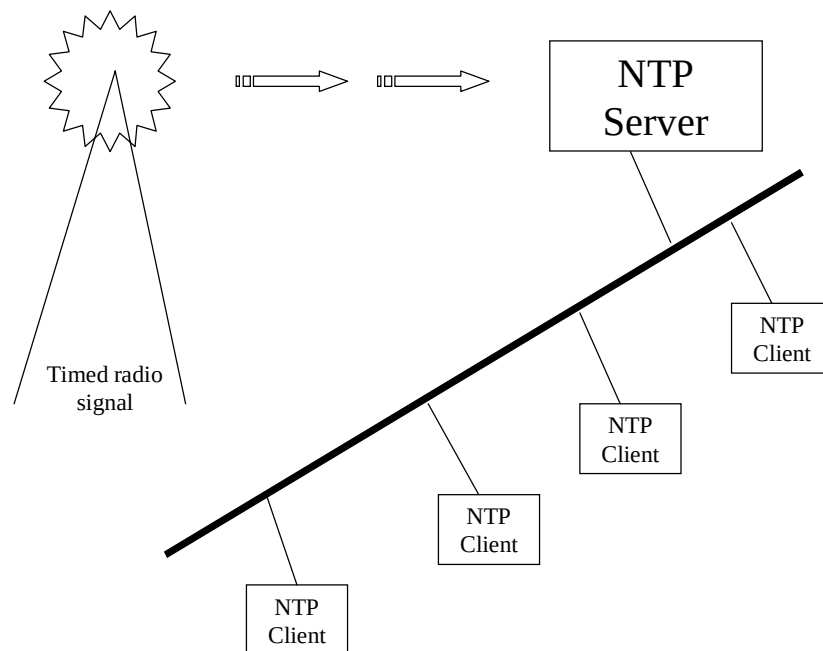


Network Time Synchronization

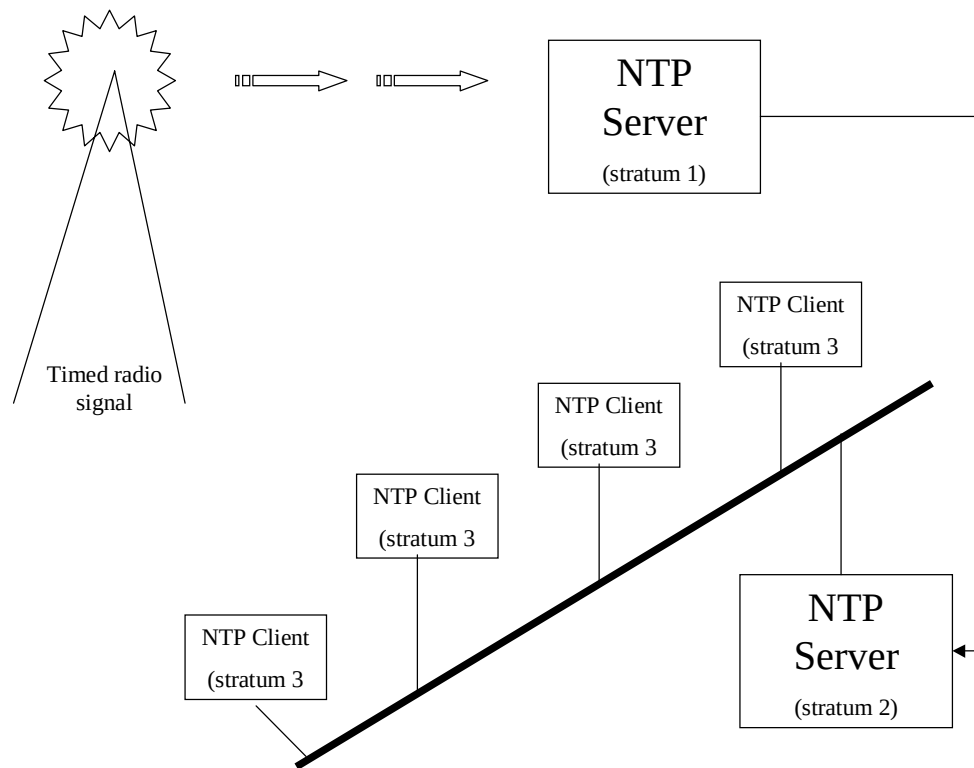
- Many network services rely on a relatively synchronized view of the time (NFS).

System A should see the current time as being within several tens of milliseconds of the current time as seen by System B.

- NTP is a sophisticated time synchronization service allowing clients to be served a more accurate view of the current time than what they currently have.
 - NTP is typically traced back to a reliable system associated with a highly accurate clock.
 - The client can choose hosts offering more accurate time over hosts offering less accurate time.



- Accuracy of time offering is determined by *stratum*.
 - Stratum is a number describing the relative accuracy of your system's time.
 - Stratum 1 servers are connected to a highly accurate time source such as a radio receiver tuned to a time-code signal broadcast by the government.
 - Stratum 2 servers get their time from stratum 1 servers, and so on.
 - A client typically chooses to set its time to that offered by the server with the lowest stratum number associated with its version of time.
- Implement NTP through the TCPIP\$NTP process (like xntpd in UNIX), which uses the SYS\$SPECIFIC:[TCPIP\$NTP]TCPIP\$NTP.CONF (like /etc/ntp.conf in UNIX) file for its configuration.



Time Sources

Time sources could be:

- **Internet NTP servers.**

- Obtain permission from the contact person listed for the Internet server before specifying it as a server for your local NTP servers.
- Get a list of available public time servers from **www.eecis.udel.edu**.
- Choose several time servers geographically near your client.
- To minimize impact on Internet stratum 1 servers, you should choose a stratum 2 server.

- Sample stratum 1 information from the web page (approximately 80 total).

bitsy.mit.edu (18.72.0.3)

Location: MIT Information Systems, Cambridge, MA

Geographic Coordinates: 42:21:36N, 71:05:24W

Synchronization: NTP primary (WWV clock), VAX/Unix

Service Area: NSFNET, NEARnet area

Access Policy: open access

Contact: Jeff Schiller (jis@mit.edu)

- Sample stratum 2 information from the web page (approximately 110 total).

black-ice.cc.vt.edu (128.173.14.71)

Location: Virginia Tech Computing Center, Blacksburg, VA, USA

Synchronization: NTP V3 secondary (Stratum 2), RS/6000-220

Service Area: southeastern US, anyplace netwise close to
ANSNet

Greensboro

Access Policy: open access

Contact: valdis@black-ice.cc.vt.edu or Valdis.Kletnieks@vt.edu

Note: ntp-2.vt.edu is a CNAME for black-ice.cc.vt.edu

- **A local reference clock.**

Select a system on your network to be the local reference clock.

Configuration Modes

You can configure NTP hosts in:

- **Client/server mode.**
 - Local host wants to obtain time from the remote server and also is willing to supply time to the remote server, if needed.
 - Appropriate in configurations involving several redundant time servers interconnected through diverse network paths.
 - Internet time servers typically use this mode.
 - Configuration file would contain:
`peer 192.206.126.4`
 - Peer is one *of equal standing*.
 - NTP expands the definition of peer to include all NTP participants.
- **Client mode.**
 - Indicates that the local host wants to obtain time from the remote server.
 - Also indicates that the node is not willing to provide time to the remote server.
 - Useful for file server and workstation clients that do not provide synchronization to other local clients.
 - A host with high stratum typically uses this mode.
 - Configuration file would contain:
`server 192.206.126.3`
- **Broadcast mode.**
 - Local server sends periodic broadcast messages to a client population at the broadcast/multicast address specified.
 - Typically applies to the local server operating as a sender.
 - Configuration file would contain:
`broadcast 192.206.126.255`

NTP Directory and Template

\$

\$ **dir sys\$specific:[tcpip\$ntp]**

Note, no configuration file!

Directory SYS\$SPECIFIC:[TCPIP\$NTP]

Template file will become configuration file.

LOGIN.COM;1	TCPIP\$NTP.LOG;71	TCPIP\$NTP.LOG;70	TCPIP\$NTP.LOG;69
TCPIP\$NTP.LOG;68	TCPIP\$NTP.LOG;67	TCPIP\$NTP.LOG;66	TCPIP\$NTP.LOG;65
<u>TCPIP\$NTP.TEMPLATE;1</u>		TCPIP\$NTP_RUN.LOG;39	
TCPIP\$NTP_RUN.LOG;38		TCPIP\$NTP_RUN.LOG;37	
TCPIP\$NTP_RUN.LOG;36		TCPIP\$NTP_RUN.LOG;35	

Total of 14 files.

\$

\$ **type sys\$specific:[tcpip\$ntp]TCPIP\$NTP.TEMPLATE**

Copyright (c) Digital Equipment Corporation, 1998

Example NTP Configuration File

Rename this template to TCPIP\$NTP.CONF.

See the DIGITAL TCP/IP Services for OpenVMS Management manual for
additional commands and detailed instructions on using this
configuration file.

The Network Time Protocol (NTP) provides synchronized timekeeping among
a set of distributed time servers and clients. The local OpenVMS host
maintains an NTP configuration file, TCPIP\$NTP.CONF, of participating peers.
TCPIP\$NTP.CONF is maintained in the SYS\$SPECIFIC:[TCPIP\$NTP] directory.

As the system manager populating this file, you must determine the
peer hosts with which the local hosts should negotiate and synchronize
Include at least one (but preferably three) hosts that you are
certain have the following characteristics:

* provide accurate time
* synchronize to Internet Time Servers (if they are not themselves
Internet Time Servers).

The NTP configuration file is not dynamic, and therefore requires
restarting NTP after being edited to make the changes take effect.
However, you can make runtime configuration requests interactively
using the TCPIP\$NTPDC utility.

NTP Template File

Your NTP configuration file should always include the following
 # driftfile entry. The driftfile is the name of the file that stores
 # the clock drift (also known as frequency error) of the system clock.

```
driftfile SYS$SPECIFIC:[TCPIP$NTP]TCPIP$NTP.DRIFT
```

Samples peer entries follow. Replace them with your own list of hosts
 # and identify the appropriate association mode. If you specify
 # multiple hosts, NTP can choose the best source with which to
 # synchronize. This also provides reliability in case one of the hosts
 # becomes unavailable.

Identify each peer with a fully-qualified DNS host name or with
 # an IP address in dotted-quad notation.

```
peer 18.72.0.3
peer 130.43.2.2
peer 16.1.0.22
peer parrot.com
```

The following commands allow interoperation of NTP with another time service
 # such as DTSS. If enabled (by removing #), NTP will not set the system clock.

```
# server 127.127.1.0 prefer
# fudge 127.127.1.0 stratum 0
$
$
$
$ set def sys$specific:[tcpip$ntp]
$
```

Create the configuration file.

```
$ copy TCPIP$NTP.TEMPLATE;1 tcpip$ntp.conf
$
```

Edit the NTP template file as necessary.

\$

\$! edit this file to reflect your servers.

\$

\$

\$ **type tcpip\$ntp.conf**

(...)

Designates that system as a client of alfman.

server 192.206.126.3 # alfman

Node cuff is a peer.

peer 192.206.126.4 # cuff

(...)

\$

\$ **rsh /user="root" alfman "date"**

Server time.

Fri Jul 2 10:06:29 EDT 1999

\$

\$ **show time**

Client time.

2-JUL-1999 10:07:46

\$

**Note**

Times are out of synchronization by over one minute!

NTP Shutdown and Startup

\$

\$ **tcpip show serv ntp/full**

NTP process details.

Service: NTP

		State:	<u>Enabled</u>	
Port:	123	Protocol:	UDP	Address: 0.0.0.0
Inactivity:	0	User_name:	<u>TCPIP\$NTP</u>	Process: <u>TCPIP\$NTP</u>
Limit:	1	Active:	0	Peak: 0

File: TCPIP\$SYSTEM:TCPIP\$NTP_RUN.COM

Flags: None

Socket Opts: Rcheck Scheck

Receive: 0 Send: 0

Log Opts: Acpt Actv Dactv Conn Error Exit Logi Logo Mdfy Rjct Tim0 Addr

File: SYS\$SPECIFIC:[TCPIP\$NTP]TCPIP\$NTP_RUN.LOG

Security

Reject msg: not defined

Accept host: 0.0.0.0

Accept netw: 0.0.0.0

\$

\$

Shut down NTP.

\$ **@sys\$startup:tcpip\$ntp_shutdown**

TCPIP\$NTP server 000002B2 stopped

\$

\$

\$ tcpip show serv ntp/full

Service: NTP

State: Disabled

Port: 123

Protocol: UDP

Address: 0.0.0.0

User_name: not defined

Process: TCPIP\$NTP

\$

\$

Start up NTP.

\$ @sys\$startup:tcpip\$ntp_startup

\$

\$ tcpip show serv ntp/full

Service: NTP

State: Enabled

Port: 123

Protocol: UDP

Address: 0.0.0.0

Inactivity: 0

User_name: TCPIP\$NTP

Process: TCPIP\$NTP

(...)

\$

\$ rsh /user="root" alfman "date"

Server time.

Fri Jul 2 10:20:18 EDT 1999

\$

\$ show time

Client time

2-JUL-1999 10:20:20

\$



Note

System files are synchronized now. The two-second difference is the approximate time between the execution of the two commands.

NTP Support Files

Drift File

NTP tries to compute the approximate time error in the frequency of the CPU clock.

- It takes a day or two of NTP activities to compute a useful value.
- NTP stores and maintains the value in the drift file.
- The drift file is used when the NTP process restarts; it knows what the last calculation was and does not have to start from scratch with its calculations.

```
$ type TCPIP$NTP.DRIFT
```

```
$
```

Empty file. No significant drift yet.

Log File

The log file is a synchronization status report file indicating:

- Synchronization occurrences and changes.
- System time adjustments.
- Time adjustment status.

```
$
```

```
$ type tcpip$ntp.log
```

```
2 Jul 10:19:12 ntpd version = 3-5.91
```

```
2 Jul 10:19:12 tickadj = 97, tick = 976, tvu_maxslew = 99231, est. hz = 1024
```

```
2 Jul 10:19:12 precision = 976 usec
```

```
2 Jul 10:19:12 read drift of 0 from SYS$SPECIFIC:[TCPIP$NTP]TCPIP$NTP.DRIFT
```

```
Node is stratum 4.
```

```
2 Jul 10:23:29 synchronized to 192.206.126.3, stratum=4
```

```
Time reset.
```

```
2 Jul 10:23:29 time reset (slew) -14.459596 sec
```

```
2 Jul 10:23:29 synchronization lost
```

```
$
```

```
$ rsh /user="root" alfman "date"
```

```
Fri Jul 2 10:31:48 EDT 1999
```

```
$ show time
```

```
2-JUL-1999 10:31:49
```

```
$
```

NTPTRACE

NTPTRACE traces a chain of servers back to stratum 1 (if possible).

Make a foreign command.

```
$ ntptrace := $sys$system:tcpip$ntptrace.exe
```

```
$
```

```
$
```

```
$ ntptrace
```

LOCALHOST:

stratum 5, offset 0.000001, synch distance 0.02396

ALFMAN:

Internet display would be more involved.

stratum 4, offset 0.004401, synch distance 0.01129

```
$
```

NTPDC

NTPDC allows querying of NTP nodes (NTPQ handles more versions).

Make a foreign command.

```
$ run sys$system:tcip$ntpd.exe
```

```
ntpd> help
```

Commands available:

addpeer	ctlstats	exit	iostats	loopinfo	preset
addserver	debug	fudge	keyid	memstats	pstats
authinfo	delrestrict	help	keytype	monlist	quit
broadcast	disable	host	leapinfo	passwd	readkeys
controlkey	enable	hostnames	listpeers	peers	requestkey
reset	sysstats	unrestrict			
reslist	timeout	untrustedkey			
restrict	timerstats	version			
showpeer	trustedkey				
sysinfo	unconfig				

System information.

```
ntpd> sysinfo
```

```
system peer:      ALFMAN
system peer mode: client
leap indicator:   00
```

Stratum value.

```
stratum:          5
precision:        -11
root distance:    0.00293 s
root dispersion:  0.02135 s
reference ID:     [192.206.126.3]
reference time:   bb274de3.8c448000  Fri, Jul  2 1999 14:40:35.547
```

system flags: auth monitor pll stats

frequency: 0.000 ppm

stability: 8.852 ppm

broadcastdelay: 0.003906 s

authdelay: 0.000122 s

ntpd> **pstats alfman**

remote host: ALFMAN

local interface: 192.206.126.27

time last received: 57s

time until next send: 7s

reachability change: 1017s

packets sent: 21

packets received: 21

bad authentication: 0

bogus origin: 0

duplicate: 0

bad dispersion: 8

bad reference time: 0

candidate order: 1

ntpd> **Listpeers**

sym_active CUFF

client ALFMAN

ntpd> **exit**

NTPQ

The NTPQ monitor program handles multiple versions of NTP—v1, 2, and 3.

Make a foreign command.

```
$ ntpq := $sys$system:tcpip$ntpq.exe
```

```
$
```

Show peer information.

```
$ ntpq -p
```

remote	refid	st	t	when	poll	reach	delay	offset	disp
*ALFMAN	LOCAL(1)	4	u	12	64	377	2.93	16.125	5.11
CUFF	0.0.0.0	16	u	-	64	0	0.00	0.000	16000.0

```
$
```

```
$
```

Interactive mode.

```
$ ntpq -i
```

```
ntpq> help
```

Commands available:

addvars	debug	keyid	lpeers	passociations
associations	exit	keytype	mreadlist	passwd
authenticate	help	lassociations	mreadvar	peers
clearvars	host	lopeers	ntpversion	status
cooked	hostnames	lpassociations	opeers	quit
raw	timeout			
readlist	version			
readvar	writelist			
rmvars	writevar			
showvars				

ntpq> **version**

Version of NTP.

ntpd version = 3-5.91

Peer information interactively.

ntpq> **peers**

remote	refid	st	t	when	poll	reach	delay	offset	disp
*ALFMAN	LOCAL(1)	4	u	63	64	377	2.93	16.125	5.11
CUFF	0.0.0.0	16	u	-	64	0	0.00	0.000	16000.0

ntpq> **exit**

\$

\$

TCPTRACE

UDP port 123 used by NTP.

\$ **tcptrace /packets=100/port=local=123/proto=udp/full**

TCPIPtrace full display XMT packet 2 at 2-JUL-1999 10:48:03.77

IP Version = 4, IHL = 5, TOS = 00, Total Length = 76 = ^x004C

IP Identifier = ^xC7D0, Flags (0=0,DF=1,MF=0),

Fragment Offset = 0 = ^x0000, Calculated Offset = 0 = ^x0000

IP TTL = 30 = ^x1E, Protocol = 17 = ^x11, Header Checksum = ^x1715

den255 (client)

IP Source Address = 192.206.126.27

alfman (server)

IP Destination Address = 192.206.126.3

UDP Source Port = 123, UDP Destination Port = 123

UDP Header and Datagram Length = 56 = ^x0038, Checksum = ^xBED8

1B7ECEC0	1517111E	0040D0C7	4C000045	0000	E..L..@.....~.
F506051B	D8BE3800	7B007B00	037ECEC0	0010	..~...{.{.8.....
634F27BB	037ECEC0	7B070000	40000000	0020	...@...{..~..'0c
634F27BB	00309ABA	634F27BB	00E02AB7	0030	.*... '0c...0..'0c
	00500DC6	A34F27BB	00E02AB7	0040	.*... '0...P.

TCPIPtrace full display RCV packet 3 at 2-JUL-1999 10:48:03.77

IP Version = 4, IHL = 5, TOS = 00, Total Length = 76 = ^x004C
IP Identifier = ^x1866, Flags (0=0,DF=0,MF=0),
Fragment Offset = 0 = ^x0000, Calculated Offset = 0 = ^x0000
IP TTL = 30 = ^x1E, Protocol = 17 = ^x11, Header Checksum = ^x0680

Response from server.

IP Source Address = 192.206.126.3

IP Destination Address = 192.206.126.27

UDP Source Port = 123, UDP Destination Port = 123

UDP Header and Datagram Length = 56 = ^x0038, Checksum = ^x5042

037ECEC0	8006111E	00006618	4C000045	0000	E..L.f.....~.
F506041C	42503800	7B007B00	1B7ECEC0	0010	..~..{.{.8PB....
814F27BB	01017F7F	E8020000	00000000	0020	'0.
A34F27BB	00500DC6	A34F27BB	00305A45	0030	EZ0..'0...P..'0.
	0020DAC9	A34F27BB	0020DAC9	0040	'0... .

(...)

Simple Network Management Protocol (SNMP)

Module 12

Introduction

This module introduces SNMP. SNMP allows a network administrator to get statistics and configuration information from the network nodes being managed. The concepts of the protocol are presented as well as configuration and utility examples. Traces of the communication between the NMS (client) and agent (server) are discussed.

Objectives

Upon completion of this module, you should be able to:

- Describe the SNMP protocol.
- Discuss SNMP concepts.
- Configure, start, and stop SNMP.
- Discuss SNMP agents, subagents, and NMS systems.
- Use SNMP test utilities.
- Trace SNMP communication.

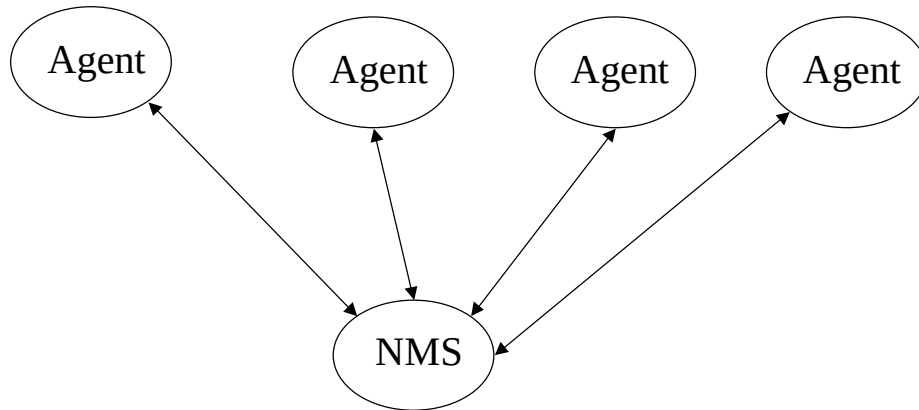
Topics

- SNMP Concepts
- Log Files and Command Procedures
- SNMP Configuration
- Using SNMP Utilities
- Tracing SNMP Protocol Interactions

Agents (Servers) and Management Stations (Clients)

SNMP is an Internet standard for exchanging network management information.

- SNMP is implemented through a client/server strategy where a network management station (NMS), the client, requests information from various agents, the servers.
- Most systems are configured as agents (servers).
- Agent server software is usually provided with TCP/IP.
- NMS client software is usually licensed separately.

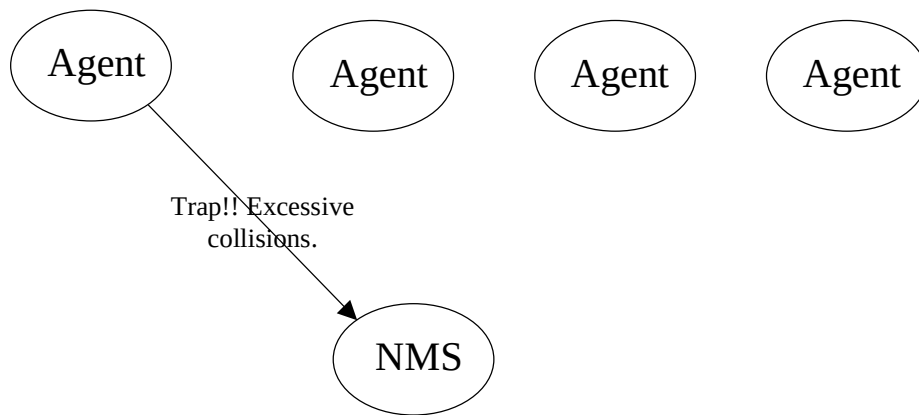


NMS Client Communicating with SNMP Server Agents

Agent (Server)

- Agent systems supply data in response to requests from the client and are:
 - Implemented at the application level over UDP using port 161.
 - Designed at the application level to support heterogeneous networks.
- Agents run on the Network Managed Entity (NME).
 - Typical NME is a router, but could be an end node or a bridge.
 - Small and easy to implement software.
 - Responds to periodic NMS *Get* requests (GetResponse).
 - May respond in real time if configured for *traps*.
- **Trap**

A *trap* is a means whereby an agent alerts the NMS of an **unusual** event. A trap is sent on UDP port 162.



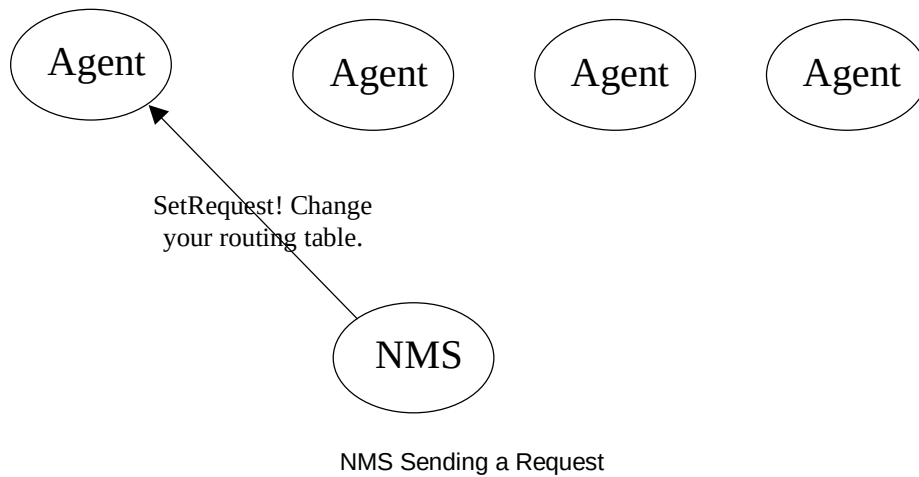
Agent Alerting NMS Through Trap Mechanism

NMS (Client)

NMS gathers information from agents and presents it to the network administrator. NMS also:

- Provides analysis tools and (usually) graphical display capability.
- Polls agents periodically (GetRequest).
- May be allowed to alter settings (SetRequest) on the agent.

The following figure shows NMS sending a request to alter settings on the agent node.



SNMP Communication

- Data being sent and data being managed must be in a format accepted by all participants.
- Data being managed is represented using an associated standard called Management Information Base (MIB), which specifies:
 - The data items to be kept and the operations allowed on each (for example, Get, Set, or Trap).
 - Network management variable names and their meanings.
- MIB divides data into eight categories:
 - System
 - Interfaces
 - ARP
 - IP
 - ICMP
 - TCP
 - UDP
 - EGP
- The MIB standard can change with little impact on SNMP software.

MIB

The following table shows some MIB variables, IDs, and categories (there are many other variables).

Variable	Category	Comment
sysUpTime (3)	System (1)	Time since boot.
IpInReceives (3)	IP (4)	Count of IP datagrams received.
IcmpInEchos (8)	ICMP (5)	Count of echo requests received.
ifNumber (1)	Interfaces (2)	Count of network interfaces.
tcpInSegs (10)	TCP (6)	Count of TCP Segments received.

You can find rules for defining MIB variables in the Structure of Management Information (SMI) standard specification.

- (Held in RFCxxxx.MY files—see SYS\$EXAMPLES.)
- Restricts variable types.
- Defines variable naming rules.

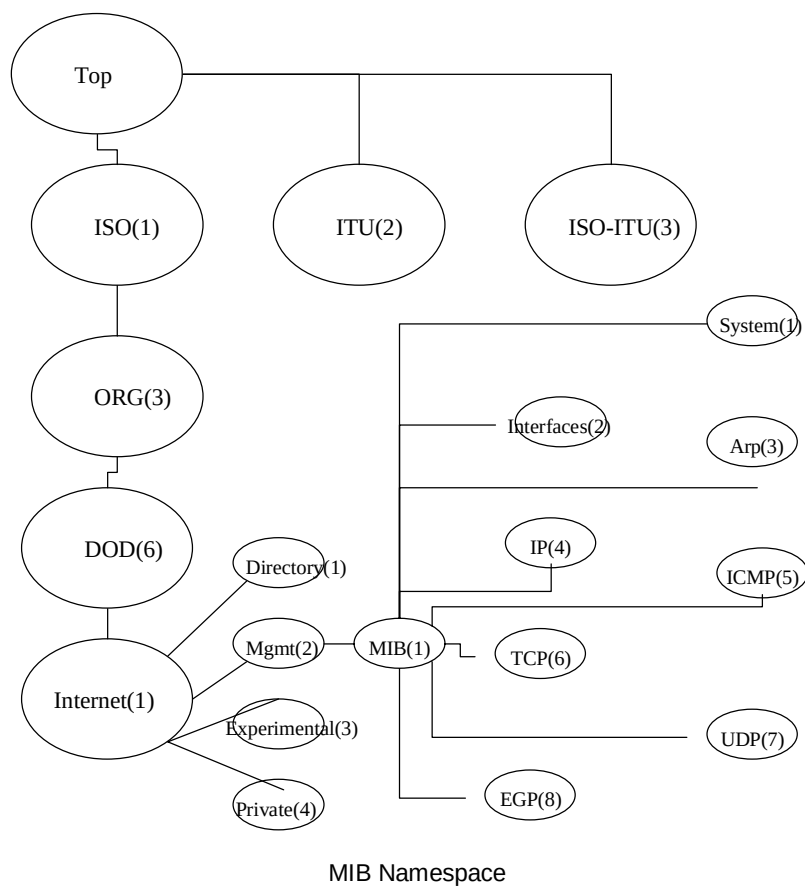
ASN.1

- Part of SMI specifies that variables must be referenced using Abstract Syntax Notation 1 (ASN.1).
- ASN.1 is a notation defined by the International Standards Organization (ISO), which:
 - Provides a human readable format.
(iso.org.dod.internet.mgmt.mib.ip.ipInReceives)
 - Provides a compact, data-encoded format for use by SNMP.
1.3.6.1.2.1.4.3
 - Provides simplicity for network management protocols.
1(iso).3(org).6(dod).1(internet).2(mgmt).1(mib).4(ip).3(ipInReceives)

Object Names and Identifiers

- MIB is not restricted to network management data.
- Namespace is hierarchical with authority for names potentially delegated out beyond the central naming authority (ISO and ITU).
 - Concept is similar to DNS.
 - (ITU is the International Telecommunication Union—formerly part of the Consultative Committee for International Telephone and Telegraph (CCITT).)
- Root (top) is unnamed in the hierarchy.

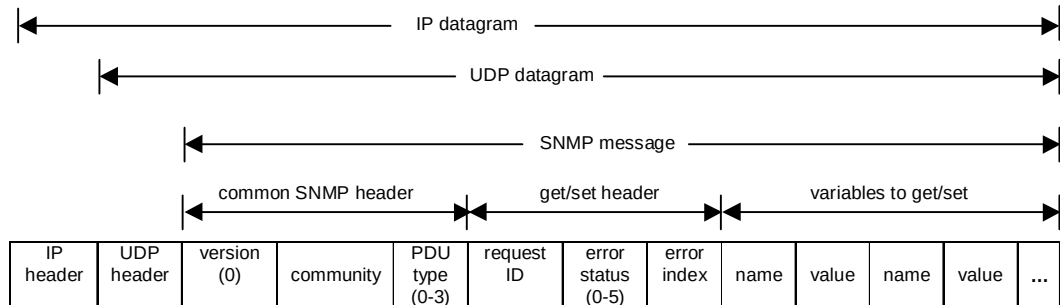
In the following figure, 1.3.6.1.2.1.4.3 represents
iso.org.dod.internet.mgmt.mib.ip.ipInReceives.



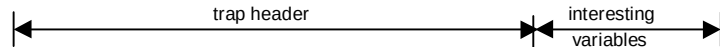
SNMP Protocol Format

The SNMP format consists of:

- SNMP protocol version number 2.
- Community ID — defines a group of commonly administered systems (like a zone in DNS).
- Data — arranged in protocol data units (PDUs), including object ID and other data.



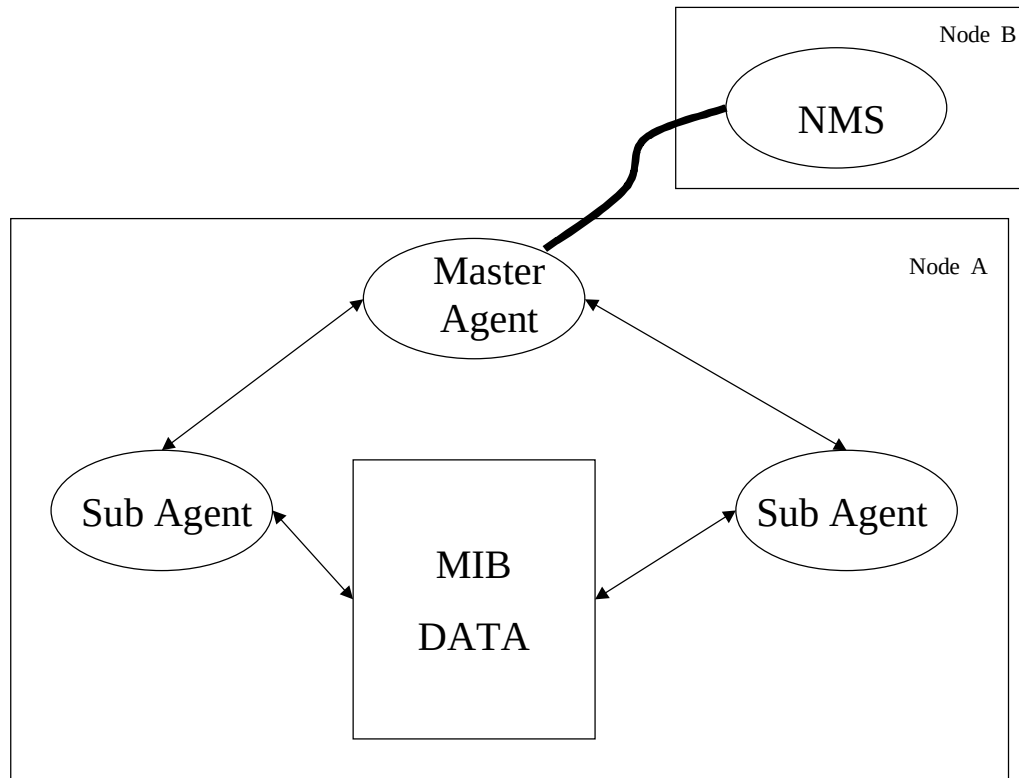
PDU type (4)	enterprise	agent addr	trap type (0-6)	specific code	time stamp	name	value	...
--------------	------------	------------	-----------------	---------------	------------	------	-------	-----



SNMP Messages Format

SNMP on OpenVMS

- SNMP on OpenVMS provides:
 - SNMP master agent
 - Three subagents (MIB II, HostResources MIB, and Operating System MIB)
 - MIB converter and compiler
 - MIB browser
 - Trap send and receive utility programs
- Each subagent contains routines that perform read and write operations on its MIB data items.
- The following figure shows NMS communicating with the master agent, which then communicates with subagents.



SNMP Components

Component	Description
Master agent SNMP Version 2	TCPIP\$ESNMP keeps track of managed objects.
MIB II	TCPIP\$OS_MIBS provides information about the TCP/IP protocol stack and other network activity.
Host resources MIB	TCPIP\$HR_MIB provides information about the host system.
MIB converter	Extracts a MIB definition in ASN.1 notation in a text file into a MIB definition (.MY) file.
MIB compiler	Compiles MIB-definition files into source code templates for use in building subagents.
MIB utility programs	Act as simple clients to obtain a set of values for a MIB and to listen for and send trap messages (snmp_request, others).

SNMP Through TCPIP\$CONFIG

- The TCPIP\$CONFIG procedure sets up:
 - SNMP at port 161.
 - SNMP master agent, at port 242 (used in the OpenVMS implementation as the master agent port number that is known to the local subagents).
 - Files in the SYS\$SYSDEVICE:[TCPIP\$SNMP] directory.
- You can run the SNMP startup procedure (SYS\$STARTUP:TCPIP\$SNMP_STARTUP.COM) from TCPIP\$STARTUP.COM or you can run it directly.
- TCPIP\$SNMP_STARTUP.COM:
 - Checks the TCP/IP Services license and enables the SNMP service.
 - Installs images with the required privileges (BYPASS, PHY_IO, and WORLD)
 - Runs TCPIP\$EXTENSION_MIB_STARTUP.COM (found in SYS\$SYSDEVICE:[TCPIP\$SNMP]), which handles user-written subagents.
 - Sends a UDP datagram to the SNMP port. When the port receives the datagram, the auxiliary server runs the TCPIP\$SNMP_RUN.COM procedure (found in TCPIP\$SYSTEM:).

- TCPIP\$SNMP_RUN.COM:
 - Purges files in SYS\$SYSDEVICE:[TCPIP\$SNMP].
 - Runs the master agent and subagent as detached processes.
 - Runs TCPIP\$EXTENSION_MIB_RUN.COM (found in SYS\$SYSDEVICE:[TCPIP\$SNMP]) to start any additional subagents.
- As each subagent starts, it contacts the master agent, which registers any MIB subtrees that the subagent maintains and communicates the port number on which it listens.
- Once SNMP has started:
 - The master agent listens for incoming SNMP requests from clients on port 161.
 - ◆ Authentication is limited to the validation of the community name.
 - ◆ When a request arrives, the master agent communicates with the subagent at the registered port.
 - The subagent collects the requested data and returns the data to the master agent.
 - The master agent responds to the client from which the original request was made.

SNMP Startup and Shutdown

- You can run TCPIP\$SNMP_SHUTDOWN.COM from TCPIP\$SHUTDOWN.COM or you can run it directly.
- TCPIP\$SNMP_SHUTDOWN.COM:
 - Stops subagent processes.
 - Runs TCPIP\$EXTENSION_MIB_SHUTDOWN.COM (found in SYS\$SYSDEVICE:[TCPIP\$SNMP]), which stops any additional subagent processes.

Log Files and Command Procedures

\$

SNMP directory.

\$ **set def sys\$sysdevice:[tcpip\$snmp]**

\$

\$

\$ **dir *.com/col=1**

Directory SYS\$SYSDEVICE:[TCPIP\$SNMP]

LOGIN.COM;1

For user written subagents.

TCPIP\$EXTENSION_MIB_RUN.COM;1

TCPIP\$EXTENSION_MIB_SHUTDOWN.COM;1

TCPIP\$EXTENSION_MIB_STARTUP.COM;1

Total of 4 files.

\$

\$

\$ **dir *.err/col=1**

Directory SYS\$SYSDEVICE:[TCPIP\$SNMP]

Master server log files (from SYS\$ERROR).

TCPIP\$ESNMP_SERVER.ERR;42

TCPIP\$ESNMP_SERVER.ERR;41

TCPIP\$ESNMP_SERVER.ERR;40

TCPIP\$ESNMP_SERVER.ERR;39

TCPIP\$ESNMP_SERVER.ERR;38

TCPIP\$ESNMP_SERVER.ERR;37

Host resources MIB subagent log files (from SYS\$ERROR).

TCPIP\$HR_MIB.ERR;1

Total of 7 files.

\$

```
$ dir *.log/col=1
```

Directory SYSSYSDEVICE:[TCPIP\$SNMP]

Master server log files (from SYSS\$OUTPUT).

TCPIP\$ESNMP_SERVER.LOG;42

(...)

TCPIP\$ESNMP_SERVER.LOG;37

Host resources MIB subagent log files (from SYSS\$OUTPUT).

TCPIP\$HR_MIB.LOG;42

(...)

TCPIP\$HR_MIB.LOG;37

MIB-II subagent log files (from SYSS\$OUTPUT).

TCPIP\$OS_MIBS.LOG;42

(...)

TCPIP\$OS_MIBS.LOG;37

Log from SNMP_RUN.COM.

TCPIP\$SNMP_RUN.LOG;42

(...)

TCPIP\$SNMP_RUN.LOG;38

Total of 23 files.

\$

TCPIP\$SNMP_RUN.LOG

\$

\$ **type TCPIP\$SNMP_RUN.LOG**

\$ SET NOON

\$ VERIFY = F\$VERIFY(F\$TRNLNM("SYLOGIN_VERIFY"))

\$ GOTO RUN_AGENTS

\$RUN_AGENTS:

\$!

\$! Default should be the SNMP service directory, but specify it and files

\$! to be safe. V1.0-01

\$ SAVE=5+2*F\$GETSYI("CLUSTER_NODES")

\$ PURGE/KEEP=5 SYS\$SYSDEVICE:[TCPIP\$SNMP]*.LOG,*.ERR

Purge log files.

\$!

\$ SET NOON ! ba1

\$! Run the master agent

\$!

\$! DEFINE/USER SYS\$OUTPUT NL:

\$! DEFINE/USER SYS\$error NL:

\$ IMAGENAME = "TCPIP\$ESNMP_SERVER"

\$ PROCNAME = "TCPIP\$ESNMP"

\$ GOSUB RUN_AGENT

\$RUN_AGENT: ! V5.0-01

\$ PROTSTRING = F\$FILE("SYS\$SYSTEM:TCPIP\$ESNMP_SERVER.EXE", "PRO")

\$ IF F\$LOCATE("WORLD=RE",PROTSTRING) .GE. F\$LENGTH(PROTSTRING)

\$ ELSE

Start the master server.

\$ RUN SYS\$SYSTEM:TCPIP\$ESNMP_SERVER -

/DETACHED -

/ERROR=SYS\$SYSDEVICE:[TCPIP\$SNMP]TCPIP\$ESNMP_SERVER.ERR -

/OUTPUT=SYS\$SYSDEVICE:[TCPIP\$SNMP]TCPIP\$ESNMP_SERVER.LOG -

/PROCESS_NAME=TCPIP\$ESNMP -

/UIC=TCPIP\$SNMP

%RUN-S-PROC_ID, identification of created process is 000000D8

\$ ENDIF

\$ RETURN

```

$! RUN SYS$SYSTEM:TCPIP$ESNMP_SERVER -
$! /DETACHED -
$! /ERROR=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$ESNMP.ERR -
$! /OUTPUT=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$ESNMP.LOG -
$! /PROCESS_NAME=TCPIP$ESNMP -
$! /UIC=TCPIP$SNMP
$!
$! Now, run the MIB-2 and HRM subagents:
$!
$! DEFINE/USER SYS$OUTPUT NL:
$! DEFINE/USER SYS$ERROR NL:
$ IMAGENAME = "TCPIP$OS_MIBS"
$ PROCNAME = "TCPIP$OS_MIBS"
$ GOSUB RUN_AGENT
$RUN_AGENT: ! V5.0-01
$ PROTSTRING = F$FILE("SYS$SYSTEM:TCPIP$OS_MIBS.EXE", "PRO")
$ IF F$LOCATE("WORLD=RE",PROTSTRING) .GE. F$LENGTH(PROTSTRING)
$ ELSE
    Start the MIB-II subagent process.
$ RUN SYS$SYSTEM:TCPIP$OS_MIBS -
    /DETACHED -
    /ERROR=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$OS_MIBS.ERR -
    /OUTPUT=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$OS_MIBS.LOG -
    /PROCESS_NAME=TCPIP$OS_MIBS -
    /UIC=TCPIP$SNMP
%RUN-S-PROC_ID, identification of created process is 000000D9
$ ENDIF
$ RETURN

```

```
$! RUN SYS$SYSTEM:TCPIP$OS_MIBS -
$! /DETACHED -
$! /ERROR=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$OS_MIBS.ERR -
$! /OUTPUT=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$OS_MIBS.LOG -
$! /PROCESS_NAME=TCPIP$OS_MIBS -
$! /UIC=TCPIP$SNMP
$!
$! DEFINE/USER SYS$OUTPUT NL:
$! DEFINE/USER SYS$error NL:
$ IMAGENAME = "TCPIP$HR_MIB"
$ PROCNAME = "TCPIP$HR_MIB"
$ GOSUB RUN_AGENT
$RUN_AGENT: ! V5.0-01
$ PROTSTRING = F$FILE("SYS$SYSTEM:TCPIP$HR_MIB.EXE", "PRO")
$ IF F$LOCATE("WORLD=RE",PROTSTRING) .GE. F$LENGTH(PROTSTRING)
$ ELSE
    Start the host resources subagent.
$ RUN SYS$SYSTEM:TCPIP$HR_MIB -
    /DETACHED -
    /ERROR=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$HR_MIB.ERR -
    /OUTPUT=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$HR_MIB.LOG -
    /PROCESS_NAME=TCPIP$HR_MIB -
    /UIC=TCPIP$SNMP
%RUN-S-PROC_ID, identification of created process is 000000DA
$ ENDIF
$ RETURN
```



```

$! RUN SYS$SYSTEM:TCPIP$HR_MIB -
$! /DETACHED -
$! /ERROR=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$HR_MIB.ERR -
$! /OUTPUT=SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$HR_MIB.LOG -
$! /PROCESS_NAME=TCPIP$HR_MIB -
$! /UIC=TCPIP$SNMP
$!
$! V5.0-01
$ ON WARNING THEN CONTINUE
$ IF F$SEARCH("SYS$SYSDEVICE:[TCPIP$SNMP]*.*") .EQS. ""
$ ELSE
$!
$! Try normal extension subagent run.
$! To include extension sub-agents in SNMP run sequence edit
$! SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$EXTENSION_MIB_RUN.COM as needed.
$! Be sure you have done normal configuration on SNMP.
$! Do not edit TCPIP$SNMP_RUN.COM.

        Hunt for user-written subagent extensions.
$!
$ ON WARNING THEN GOTO NO_EXTENSION_FILE ! V5.0-01
$ @SYS$SYSDEVICE:[TCPIP$SNMP]TCPIP$EXTENSION_MIB_RUN.COM ! V5.0-00
$ GOTO BEGIN
$BEGIN:
$!
$ ON CONTROL_Y THEN GOTO EXIT
$! Edit RUN commands as needed. Remove the GOTO

```

\$!

\$ GOTO EXIT

\$EXIT:

\$ EXIT

\$ ENDIF

\$!

\$EXIT:

\$ SET ON ! ba1

\$ EXIT

TCPIP\$SNMP job terminated at 30-JUN-1999 16:04:56.92

Accounting information:

Buffered I/O count:	115	Peak working set size:	1744
---------------------	-----	------------------------	------

Direct I/O count:	75	Peak virtual size:	167808
-------------------	----	--------------------	--------

Page faults:	575	Mounted volumes:	0
--------------	-----	------------------	---

Charged CPU time:	0 00:00:00.40	Elapsed time:	0 00:00:01.70
-------------------	---------------	---------------	---------------

\$

Changing Default Configuration

```
$ tcpip show config snmp /full
```

SNMP Configuration

Flags: AuthenTraps Sets

Contact: system

Location

First: Manchester, MA

Community	Type	Address_list
public	Read	0.0.0.0

\$

```
$ dir *.dat
```

Directory SYS\$SYSDEVICE:[TCPIP\$SNMP]

Configuration file.

```
TCPIP$SNMP_CONF.DAT;42
```

```
TCPIP$SNMP_CONF.DAT;41 TCPIP$SNMP_CONF.DAT;40
TCPIP$SNMP_CONF.DAT;39
```

Total of 4 files.

\$

```
$ type TCPIP$SNMP_CONF.DAT;42
```

Public community has read access.

```
community public 0.0.0.0 read
```

\$

```
$ tcpip
```

```
TCPIP> set config snmp /contact="Dennis O'Brien"
```

```
TCPIP>
```

```
TCPIP> set config snmp /location=first="Manchester, MA"
```

```
TCPIP>
```

NMS address.

TCPIP> set config snmp /addr=192.206.126.27

TCPIP>

Community name is Bruden.

Address 0.0.0.0 means any host.

TCPIP> set config snmp /community="Bruden" /addr=0.0.0.0

TCPIP>

TCPIP> set config snmp /community="Bruden" /flags=sets

Qualifier /flags=sets allows writes also.

TCPIP>

Showing Configuration

TCPIP>

TCPIP> **sho config smp/full**

SNMP Configuration

Flags: AuthenTraps Sets

Contact: Dennis O'Brien

Location

First: Manchester, MA

Community	Type	Address_list
public	Read	0.0.0.0
Bruden	Read Write	0.0.0.0

TCPIP>

TCPIP>

TCPIP> **exit**

\$

\$

\$ **sho sys**

OpenVMS V7.2 on node DEN255 30-JUN-1999 16:43:35.49 Uptime 0 01:04:50

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
00000081	SWAPPER	HIB	16	0	0 00:00:03.16	0	0

(...)

000000D8	<u>TCPIP\$ESNMP</u>	LEF	9	664	0 00:00:00.53	260	212
----------	---------------------	-----	---	-----	---------------	-----	-----

Master agent.

000000D9	<u>TCPIP\$OS_MIBS</u>	LEF	8	551	0 00:00:00.44	257	152
----------	-----------------------	-----	---	-----	---------------	-----	-----

Subagents.

000000DA	<u>TCPIP\$HR_MIB</u>	LEF	8	536	0 00:00:01.82	270	185
----------	----------------------	-----	---	-----	---------------	-----	-----

\$

\$

Support Commands

GetRequests

\$

\$ **sho sym snmp_request**

%DCL-W-UNDSYM, undefined symbol - check validity and spelling

\$

Sets up foreign commands.

\$ **@sys\$manager:tcpip\$define_commands**

\$

\$

\$ **sho sym snmp_request**

Poor man's NMS.

SNMP_REQUEST == "\$SYS\$SYSTEM:TCPIP\$SNMP_REQUEST.EXE"

\$

\$

Gets contact name.

\$ **snmp_request den255 "Bruden" get 1.3.6.1.2.1.1.4.0**

No reply! Restart SNMP.

\$

\$

Shut down SNMP.

\$ **@sys\$startup:TCPIP\$SNMP_SHUTDOWN.COM**

TCPIP\$ESNMP stopped

TCPIP\$OS_MIBS stopped

TCPIP\$HR_MIB stopped

\$

\$

Restart SNMP.

```
$ @sys$startup:TCPIP$SNMP_STARTUP.COM
```

```
$
```

```
$
```

Gets the contact name on den255.

```
$ snmp_request den255 "Bruden" get 1.3.6.1.2.1.1.4.0
```

```
1.3.6.1.2.1.1.4.0 = Dennis O'Brien
```

```
$
```

```
$
```

Gets the contact name on alfman.

```
$ snmp_request alfman "Bruden" get 1.3.6.1.2.1.1.4.0
```

```
1.3.6.1.2.1.1.4.0 = DOB
```

```
$
```

```
$ rsh/user=root /pass="Bruden" alfman "grep DOB /etc/snmpd.conf"
```

Accesses the UNIX configuration file on alfman.

```
sysContact      DOB
```

```
$
```

```
$ rsh /user=root alfman "cat /etc/snmpd.conf"
```

```
#
```

```
(...)
```

```
#
```

```
sysName      alfman
```

```
sysLocation  Harbor St. (BR)
```

```
sysContact      DOB
```

```
(...)
```

```
snmpEnableAuthenTraps  1
```

```
(...)
```

```
community      Bruden      0.0.0.0      write
```

```
(...)
```

```
trap  Bruden      192.206.126.27
```

Note how the community name acts like a password.

```
$
```

SetRequests

\$

\$

Sets contact name on alfman.

\$ snmp_request alfman "Bruden" set 1.3.6.1.2.1.1.4.0 "-D" "Dennis O'B"

1.3.6.1.2.1.1.4.0 = Dennis O'B

\$

\$

\$ rsh/user=root alfman "grep Dennis /etc/snmpd.conf"

Changed on the UNIX side!

sysContact Dennis O'B

TCPTRACE of GetRequest

\$

\$ tcptrace

Transmits from NMS.

Requests contact name.

TCPIPtrace XMT packet 1 at 30-JUN-1999 16:56:22.82

	IP Address	Port
Source	192.206.126.27	1050
Note the SNMP port 161.		
Destination	192.206.126.3	<u>161</u>
	Packet Length	77

UDP

Note the community.

	Hex	Count	Ascii
-----	-----	-----	-----
1B7ECEC0 939C111E 00405142 4D000045 0000			E..MBQ@.....~.
2D008230 B3FE3900 A1001A04 037ECEC0 0010			..~.....9..0..-
7B040220 A06E6564 75724206 04000102 0020			 <u>Bruden</u> . ..{
00823010 00823000 01020001 02BDC1E9 0030			0...0..
ASN ...6.1.2.1.1.4.			
00 05000401 01020106 2B08060C 0040			...+.....

RCV from server.

Contains the response "Dennis O'B."

TCPIPtrace RCV packet 2 at 30-JUN-1999 16:56:22.82

	IP Address	Port
Source	192.206.126.3	<u>161</u>
Destination	192.206.126.27	1050

	Packet Length	81
UDP		

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
037ECEC0	BE0F111E	0000230F	51000045	0000	E..Q.#.....~.
01023330	C4083D00	1A04A100	1B7ECEC0	0010	..~.....=.03..
C1E97B04	0226A26E	65647572	42060400	0020	...Bruden.&...{..
<u>01062B08</u>	06163018	30000102	000102BD	0030	0.0...+..
274F2073	696E6E65	440A0400	<u>04010102</u>	0040	 <u>Dennis O'</u>
			42	0050	<u>B</u>

TCPTRACE of SetRequest

Transmitting a SET request.

TCPIPtrace XMT packet 3 at 30-JUN-1999 16:56:25.38

	IP Address	Port
Source	192.206.126.27	1051
Destination	192.206.126.3	<u>161</u>

	Packet Length
UDP	87

	Hex	Count	Ascii
-----	-----	-----	-----
1B7ECEC0	889C111E	00405242	57000045
0000			E..WBR@.....~.
37008230	2EFE4300	A1001B04	037ECEC0
0010			..~.....C..0..7
7B04022A	A36E6564	75724206	04000102
0020			Bruden.*..{
0082301A	00823000	01020001	02BDC1E9
0030			0...0..
6E65440A	04000401	01020106	2B080616
0040			...+.....Den
	42274F	2073696E	0050
			<u>nis</u> 0'B

Response from the SET request.

TCPIPtrace RCV packet 4 at 30-JUN-1999 16:56:25.50

	IP Address	Port
Source	192.206.126.3	<u>161</u>
Destination	192.206.126.27	1051

	Packet Length
UDP	81

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
037ECEC0	BD0F111E	0000240F	51000045	0000	E..Q.\$.....~.
01023330	C3083D00	1B04A100	1B7ECEC0	0010	..~.....=.03..
C1E97B04	0226A26E	65647572	42060400	0020	...Bruden.&...{..
<u>01062B08</u>	06163018	30000102	000102BD	0030	0.0...+..
274F2073	696E6E65	440A0400	<u>04010102</u>	0040	 <u>Dennis 0'</u>
			42	0050	<u>B</u>

MIB-II

The following is a portion of **RFCxxxx.MY** (from SYS\$EXAMPLES), where 1.3.6.1.2.1 = iso.org.dod.internet.mgmt.MIB.

-- groups in MIB-II

Note the ID numbers.

system OBJECT IDENTIFIER ::= { mib-2 1 }

interfaces OBJECT IDENTIFIER ::= { mib-2 2 }

at OBJECT IDENTIFIER ::= { mib-2 3 }

ip OBJECT IDENTIFIER ::= { mib-2 4 }

icmp OBJECT IDENTIFIER ::= { mib-2 5 }

tcp OBJECT IDENTIFIER ::= { mib-2 6 }

udp OBJECT IDENTIFIER ::= { mib-2 7 }

egp OBJECT IDENTIFIER ::= { mib-2 8 }

-- historical (some say hysterical)

-- cmot OBJECT IDENTIFIER ::= { mib-2 9 }

transmission OBJECT IDENTIFIER ::= { mib-2 10 }

snmp OBJECT IDENTIFIER ::= { mib-2 11 }

ID number 1 under 1.3.6.1.2.1.

-- the System group

-- Implementation of the System group is mandatory for all
 -- systems. If an agent is not configured to have a value
 -- for any of these variables, a string of length 0 is
 -- returned.

sysDescr OBJECT-TYPE

SYNTAX DisplayString (SIZE (0..255))
ACCESS read-only
STATUS mandatory

DESCRIPTION

"A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software. It is mandatory that this only contain printable ASCII characters."

Note ID number 1 under 1.3.6.1.2.1.1.

::= { system 1 }

sysObjectID OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER
ACCESS read-only
STATUS mandatory

DESCRIPTION

"The vendor's authoritative identification of the network management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree (1.3.6.1.4.1) and provides an easy and unambiguous means for determining 'what kind of box' is being managed. For example, if vendor 'Flintstones, Inc.' was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its 'Fred Router'."

::= { system 2 }

sysUpTime OBJECT-TYPE

SYNTAX TimeTicks

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The time (in hundredths of a second) since the network management portion of the system was last re-initialized."

::= { system 3 }

sysContact OBJECT-TYPE

SYNTAX DisplayString (SIZE (0..255))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The textual identification of the contact person for this managed node, together with information on how to contact this person."

::= { system 4 }

sysName OBJECT-TYPE

SYNTAX DisplayString (SIZE (0..255))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"An administratively-assigned name for this managed node. By convention, this is the node's fully-qualified domain name."

::= { system 5 }

sysLocation OBJECT-TYPE

SYNTAX DisplayString (SIZE (0..255))

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The physical location of this node (e.g.,
`telephone closet, 3rd floor')."

::= { system 6 }

sysServices OBJECT-TYPE

SYNTAX INTEGER (0..127)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value which indicates the set of services that
this entity primarily offers.

The value is a sum. This sum initially takes the value zero. Then, for each layer, L, in the range 1 through 7, that this node performs transactions for, 2^{L-1} is added to the sum. For example, a node which performs primarily routing functions would have a value of 4 (2^{3-1}). In contrast, a node which is a host offering application services would have a value of 72 ($2^{4-1} + 2^{7-1}$). Note that in the context of the Internet suite of protocols, values should be calculated accordingly:

layer	functionality
1	physical (e.g., repeaters)
2	datalink/subnetwork (e.g., bridges)
3	internet (e.g., IP gateways)
4	end-to-end (e.g., IP hosts)
7	applications (e.g., mail relays)

For systems including OSI protocols, layers 5 and 6 may also be counted."

::= { system 7 }

Introduction

This module introduces several options for remote printing. The focus is on the LPR/LPD option supported by TCP/IP Services for OpenVMS. This option should be familiar to UNIX administrators, although the support is different enough to merit some attention. Concept, configuration, and management of remote print queues are discussed. A brief conceptual overview of TELNETSYM and PC-NFS print options is also covered.

Objectives

Upon completion of this module, you should be able to:

- Discuss remote printing options.
- Configure and manage remote LPR/LPD printing.
- Trace print packets.
- Discuss TELNETSYM.
- Describe PC-NFS.

Topics

- Remote Printing Options
- LPR/LPD Local Configuration
- LPR/LPD Remote Configuration
- Tracing Print Packets
- TELNETSYM Overview
- PC-NFS Overview

Remote Printing Options

- There are three printing options:
 - **LPR/LPD** — UNIX like commands providing local and remote printing.
 - **Telnet Print Symbiont (TELNETSYM)** — provides standard OpenVMS print features using a terminal server or network printer.
 - **PC-NFS** — provides printer access to PCs running the PC-NFS software.
- Printing services are handled through client/server communication between a command (**PRINT**) and a symbiont (**TELNETSYM**, **PRINTSYM**) or a printing daemon process (**LPD**).
 - You can extend the communication path to access printing services on remote nodes.
 - A client would issue a print request on one node, and the server would handle the actual printing.

LPR/LPD

- TCP/IP Services for OpenVMS provides UNIX like commands to set up local and remote printing.
 - PRINT — issues a print request (like the UNIX `lpr` command).
 - LPD — handles the print request.
- Printer characteristics are held in an ASCII data file `TCPIP$PRINTCAP.DAT`, found in `SYSS$SPECIFIC:[TCPIP$LPD]`.
- Manage printer characteristics through `LPRSETUP` (`SYSS$SYSTEM:TCPIP$LPRSETUP.EXE`).

Configuring a Local Printer

The PRINTCAP file has many changeable characteristics.

\$

\$

\$ dir sys\$specific:[tcpip\$lpd]

Directory SYS\$SPECIFIC:[TCPIP\$LPD]

Printer capabilities file.

LOGIN.COM;1 TCPIP\$PRINTCAP.DAT;1

Total of 2 files.

\$

\$

\$ type SYS\$SPECIFIC:[TCPIP\$LPD]TCPIP\$PRINTCAP.DAT

#

LOCAL PRINTERS

#

TCPIP\$LPD_QUEUE:\

Default printcap information.

:lp=TCPIP\$LPD_QUEUE:\

:sd=TCPIP\$LPD_SPOOL:

\$

\$

\$

LPRSETUP helps configure LPR/LPD printing.

\$ run sys\$system:tcpip\$lprsetup

TCPIP Printer Setup Program

Add printer.

Command < add delete view help exit >: **a**

Adding printer entry, type '?' for help.

Enter printer name to add : ?

Question mark (?) in response to a prompt produces help.

Enter a printer name, ie., the name by which you want to identify the printer through the lpr command, example: lpr -Pprintername.

Queue name.

Enter printer name to add : **oldnslow**

Enter the FULL name of one of the following printer types:

remote local : ?

You can set up 2 kinds of printers

'local' or 'remote'.

The default printer type is 'local'.

Enter the FULL name of one of the following printer types:

remote local : **local**

Alternate queue name.

Enter printer synonym: **ons**

Enter printer synonym:

Enter full file specification for spool directory

SPOOLER DIRECTORY 'sd' : [SYSS\$SPECIFIC:[TCPIP\$LPD.OLDNSLOW]] ?

Enter full file specification for printer log file.

printer error log file 'lf' [SYSS\$SPECIFIC:[TCPIP\$LPD]OLDNSLOW.LOG] ?

Enter the name of the printcap symbol you wish to modify. Other valid entry is :

'q' to quit (no more changes)

The names of the printcap symbols are:

sd for the printer spool directory

lf for the printer error log file

lp for the name of the local printer

ps for the LPD PrintServer extensions flag

Printcap symbols.

rm for the name of the remote host
 rp for the name of the remote printer
 fm for the printer form field
 pa for the /PASSALL flag
 nd for the /NODELETE flag
 cr for the cr flag
 p1-p8 for the /PARAMETER=(p1,...,p8) field

Enter symbol name: **p**

	Symbol	type	value
	-----	----	-----
Error log file	: lf	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.LOG
Printer Queue	: lp	STR	OLDNSLOW
Spool Directory	: sd	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW
Parameter fields	: p1	STR	

Enter symbol name: ?

Enter the name of the printcap symbol you wish to modify. Other
 valid entry is :

'q' to quit (no more changes)

Enter symbol name: **lp**

Enter a new value for symbol 'lp'? [OLDNSLOW]

Enter symbol name: **q**

	Symbol	type	value
	-----	----	-----
Error log file	: lf	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.LOG
Printer Queue	: lp	STR	OLDNSLOW
Spool Directory	: sd	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW
Parameter fields	: p1	STR	

Are these the final values for printer OLDNSLOW ? [y]

Adding comments to printcap file for new printer, type '?' for help.

Do you want to add comments to the printcap file [n] ? :

```
*****
* TCPIP$LPD_STARTUP.COM      TCPIP$LPD_PRINTCAP*
*   and TCPIP$LPD_SHUTDOWN.COM      *
*   have been updated for this printer      *
*                                           *
* Set up activity is complete for this printer*
*****
```

Command < add delete view help exit >: **ex**

\$

```

$
$ type SYSS$SPECIFIC:[TCPIP$LPD]TCPIP$PRINTCAP.DAT
#
# LOCAL PRINTERS
#
TCPIP$LPD_QUEUE:\
    :lp=TCPIP$LPD_QUEUE:\
    :sd=TCPIP$LPD_SPOOL:
    New entry.
OLDNSLOW|oldnslow|ons:\
    :lf=/SYSS$SPECIFIC/TCPIP$LPD/OLDNSLOW.LOG:\
    :lp=OLDNSLOW\
    :sd=/SYSS$SPECIFIC/TCPIP$LPD/OLDNSLOW:
#
$ dir SYSS$SPECIFIC:[TCPIP$LPD]

Directory SYSS$SPECIFIC:[TCPIP$LPD]
    New directory for print queue.
LOGIN.COM;1      OLDNSLOW.DIR;1
TCPIP$PRINTCAP.DAT;1
TCPIP$PRINTCAP.DATLOG;1

Total of 4 files.
$
$
$ @sys$startup:tcip$config
    (...)

```

DIGITAL TCP/IP Services for OpenVMS CLIENT Components Configuration Menu

Configuration options:

1 - FTP Enabled

Printing enabled.

2 - LPR/LPD Enabled

3 - NFS Client Enabled

4 - REXEC and RSH Enabled

5 - RLOGIN Enabled

6 - SMTP Enabled

7 - TELNET Enabled

A - Configure options 1 - 7

[E] - Exit menu

(...)

DIGITAL TCP/IP Services for OpenVMS SERVER Components Configuration Menu

Configuration options:

1 - BIND Disabled

2 - BOOTP Enabled

3 - TFTP Enabled

4 - FTP Enabled

Server printing enabled.

5 - LPR/LPD Enabled

6 - NFS Enabled

7 - PC-NFS Disabled

(...)

Starting Up the Local Printer

\$

\$ @sys\$manager:tcpip\$lpd_startup

%JBC-E-JOBQUEDIS, system job queue manager is not running

%JBC-E-JOBQUEDIS, system job queue manager is not running

\$

Start the queue manager.

\$ start/queue/manager/new

\$ sho sys

OpenVMS V7.2 on node DEN255 1-JUL-1999 15:40:26.55 Uptime 0 02:06:54

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
(...)							
000000AD	QUEUE_MANAGER	HIB	8	107	0 00:00:00.20	131	168
000000B0	SYMBIONT_2	HIB	4	7	0 00:00:00.03	73	42

\$

Associate a serial port with the queue.

\$ init/queue/start/on=tta0: oldie

\$

Deleting the Printer

\$

\$ **run sys\$system:tcpip\$lpsetup**

TCPIP Printer Setup Program

Delete a printer request.

Command < add delete view help exit >: **d**

Deleting a printer entry, type '?' for help.

View printers.

Enter printer name to delete (or view to view printcap file): **v**

#

LOCAL PRINTERS

#

TCPIP\$LPD_QUEUE:\

:lp=TCPIP\$LPD_QUEUE:\

:sd=TCPIP\$LPD_SPOOL:

OLDNSLOW|oldnslow|ons:\

:lf=/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.LOG:\

:lp=OLDNSLOW:\

:sd=/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW:

#

Enter printer name to delete (or view to view printcap file): **oldnslow**

	Symbol	type	value
	-----	----	-----
Error log file	: lf	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.LOG
Printer Queue	: lp	STR	TTA0
Spool Directory	: sd	STR	/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW

Delete oldnslow, are you sure? [n] **y**

Couldn't unlink lf file: /SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.LOG

couldn't delete files from spooler directory (/SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW)

%RMS-E-FNF, file not found

Removed spooling directory: /SYS\$SPECIFIC/TCPIP\$LPD/OLDNSLOW.DIR

Command < add delete view help exit >: **e**

Reconfiguring the Local Printer

```
$ run sys$system:tcpip$lpsetup
```

TCPIP Printer Setup Program

Command < add delete view help exit >: **a**

Adding printer entry, type '?' for help.

Recreate a printer with a name matching the queue name.

Enter printer name to add: **oldie**

From: \$ init/queue/start/on=tta0: oldie

Enter the FULL name of one of the following printer types:

remote local : local

(...)

\$

```
$ sho queue oldie
```

Terminal queue OLDIE, idle, on DEN255::TTA0:, mounted form DEFAULT

\$

Physical connection works.

```
$ type dka300:[sys0.tcpipcourse]temp.txt/out=tta0:
```

\$

Start the printer.

```
$ @sys$manager:tcpip$lpd_startup
```

\$

```
$ sho sys
```

OpenVMS V7.2 on node DEN255 1-JUL-1999 15:55:55.07 Uptime 0 02:22:23

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
(...)							
000000AD	QUEUE_MANAGER	HIB	9	123	0 00:00:00.25	131	168
000000B0	SYMBIONT_2	HIB	4	7	0 00:00:00.03	73	42
000000B1	TCPIP\$LPD_QUEUE	HIB	4	63	0 00:00:00.17	245	118

\$

```
$ sho dev tta0:/full
```

Terminal TTA0:, device type unknown, is online, allocated, record-oriented device, carriage control.

Error count	0	Operations completed	12
Owner process	" <u>SYMBIONT_2</u> "	Owner UIC	[SYSTEM]
Owner process ID	000000B0	Dev Prot	S:RWPL,0:RWPL,G,W
Reference count	2	Default buffer size	80

```
$
```

```
$ print/queue=oldie dka300:[sys0.tcpipcourse]temp.txt
```

Works fine.

Job TEMP (queue OLDIE, entry 2) started on OLDIE

```
$
```

```
$
```

```
$ sho queue oldie
```

Terminal queue OLDIE, idle, on DEN255::TTA0:, mounted form DEFAULT

```
$
```

```
$
```

UNIX style queue command.

```
$ lpq oldie
```

no entries

```
$
```

Configuring the Remote Printer

\$

\$ @sys\$manager:tcpip\$lpd_shutdown

\$

\$

\$ run sys\$system:tcpip\$lprsetup

TCPIP Printer Setup Program

Command < add delete view help exit >: **a**

Adding printer entry, type '?' for help.

Enter printer name to add : ?

Enter a printer name, ie., the name by which you want to identify the printer through the lpr command, example: lpr -Pprintername.

Local name for remote printer.

Enter printer name to add : **alfp**

Enter the FULL name of one of the following printer types:

Setting up the remote printer.

remote local : **remote**

Enter printer synonym:

Enter full file specification for spool director

SPOOLER DIRECTORY 'sd' : [SYS\$SPECIFIC:[TCPIP\$LPD.ALFP]] ?

Set LPD PrintServer extensions flag 'ps' [] ?

Remote node name.

Set remote system name 'rm' [] ? **alfman**

Set remote system printer name 'rp' [] ? ?

The 'rp' parameter specifies the remote printer name argument.

The name specified must be one of the recognized names for the printer on the remote machine.

Enter a new value, or press RETURN to use the default.

Printer name on remote node.

Enter a value for 'rp'? [] ? **alfprint**

Set printer error log file 'lf' [SYS\$SPECIFIC:[TCPIP\$LPD]ALFP.LOG] ?

Enter the name of the printcap symbol you wish to modify.

(...)

Enter symbol name: **q**

	Symbol	type	value
	-----	----	-----
Error log file	: lf	STR	/SYS\$SPECIFIC/TCPIP\$LPD/ALFP.LOG
Printer Queue	: lp	STR	ALFP
PS extensions flag:	ps	STR	

Remote node.

Remote Host	: rm	STR	alfman
-------------	------	-----	--------

Remote printer.

Remote Printer	: rp	STR	alfprint
Spool Directory	: sd	STR	/SYS\$SPECIFIC/TCPIP\$LPD/ALFP

Are these the final values for printer ALFP ? [y]

Adding comments to printcap file for new printer, type '?' for help.

Do you want to add comments to the printcap file [n] ? :

Wastes paper.

Do you want the queue to default to print flag pages [y] : **n**

Start it for me.

Do you want this procedure to start the queue [n] : **y**

Creating execution queue: ALFP

Updating TCPIP\$LPD_STARTUP.COM

Updating TCPIP\$LPD_SHUTDOWN.COM

```
*****
* TCPIP$LPD_STARTUP.COM      TCPIP$LPD_PRINTCAP*
*   and TCPIP$LPD_SHUTDOWN.COM      *
*   have been updated for this printer      *
*                                           *
* Set up activity is complete for this printer*
*****
```

Command < add delete view help exit >: **e**

\$

Remote print request.

\$ **print/queue=alfp dka300:[sys0.tcpipcourse]temp.txt**

Job TEMP (queue ALFP, entry 5) started on ALFP

\$

\$ **sho queue alfp/full**

Server queue ALFP, idle, on DEN255::, mounted form DEFAULT

/BASE_PRIORITY=4 /DEFAULT=(FEED,FORM=DEFAULT) /OWNER=[SYSTEM]

/PROCESSOR=TCPIP\$LPD_SMB /PROTECTION=(S:M,O:D,G:R,W:S)

Entry	Jobname	Username	Blocks	Status
----	-----	-----	-----	-----
4	TEMP	SYSTEM	1	Holding until 1-JUL-1999 16:27:01
	Submitted	1-JUL-1999 16:22:00.95	/FORM=DEFAULT	/PRIORITY=100
	File:	_DEN255\$DKA300:[SYS0.TCIPPCOURSE]TEMP.TXT;1		

\$

UNIX style command to remove print jobs.

\$ **lprm**

_Queue name: **alfp**

\$

\$ **sho queue alfp/full**

Server queue ALFP, idle, on DEN255::, mounted form DEFAULT

/BASE_PRIORITY=4 /DEFAULT=(FEED,FORM=DEFAULT) /OWNER=[SYSTEM]

/PROCESSOR=TCPIP\$LPD_SMB /PROTECTION=(S:M,O:D,G:R,W:S)



Important

The printer on alfman must be configured at this point.

```
$
$ type sys$specific:[tcpip$lpd]tcpip$printcap.dat
#
# LOCAL PRINTERS
#
TCPIP$LPD_QUEUE:\
    :lp=TCPIP$LPD_QUEUE:\
    :sd=TCPIP$LPD_SPOOL:
#
OLDIE|oldie:\
    :lf=/SYS$SPECIFIC/TCPIP$LPD/OLDIE.LOG:\
    :lp=OLDIE:\
    :sd=/SYS$SPECIFIC/TCPIP$LPD/OLDIE:
#
ALFP|alfp:\
    :lf=/SYS$SPECIFIC/TCPIP$LPD/ALFP.LOG:\
    :lp=ALFP:\
        Remote printer in printcap file.
:rm=alfman:\
    :rp=alfprint:\
    :sd=/SYS$SPECIFIC/TCPIP$LPD/ALFP:
#
$
```

Remote Printing

\$

\$ rsh/user="root" alfman "/usr/sbin/lpc stat"

Status of alfprint printer from the perspective of alfman.

alfprint:

printer is on device '/dev/tty00' speed 9600

queuing is enabled

printing is enabled

no entries

no daemon present

\$

\$ print/queue=alfp dka300:[sys0.tcpipcourse]temp.txt

Job TEMP (queue ALFP, entry 8) started on ALFP

\$

\$ rsh/user="root" alfman "/usr/sbin/lpc stat"

alfprint:

printer is on device '/dev/tty00' speed 9600

queuing is enabled

printing is enabled

Sent from den255 (client node).

1 entry in spool area

Tue Jul 6 17:49:47 1999: Attempting to print dfA007den255.bruden.com

\$

\$ **tcpip**

TCPIP>

TCPIP> **sho serv lpd/full**

Note port number 515.

Service: LPD

State: Disabled

Port:	<u>515</u>	Protocol: TCP	Address: 0.0.0.0
		User_name: not defined	Process: TCPIP\$LPD

TCPIP>

TCPIP>

TCPIP>

TCPIP> **exit**

\$ **print/queue=alfp dka300:[sys0.tcpipcourse]temp.txt**

Job TEMP (queue ALFP, entry 9) started on ALFP

\$

\$

\$ **type dka300:[sys0.tcpipcourse]temp.txt**

Data to be printed...

\$

\$

TCPTRACE

\$ tcptrace /packets=30

TCPIPtrace XMT packet 2 at 1-JUL-1999 17:53:44.62

	IP Address	Port	Seq #	Ack #
Source	<u>192.206.126.27</u>	1023	876736001	0

From den255 to print service port (515) on alfman.

Destination	<u>192.206.126.3</u>	<u>515</u>
-------------	----------------------	------------

	Packet Length	48
TCP flags	SYN	
window	32768	

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
1B7ECEC0	4F80063C	0000BD80	30000045	0000	E..0....<..0..~.
00000000	01EE4134	0302FF03	037ECEC0	0010	..~.....4A.....
00030301	B4050402	00001E5E	00800270	0020	p...^.....

TCPIPtrace RCV packet 3 at 1-JUL-1999 17:53:44.62

	IP Address	Port	Seq #	Ack #
Source	<u>192.206.126.3</u>	<u>515</u>	4294966272	876736002

Response from alfman.

Destination	<u>192.206.126.27</u>	1023
-------------	-----------------------	------

	Packet Length	48
TCP flags	SYN ACK	
window	33580	

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
037ECEC0	7EA9063C	00408E17	30000045	0000	E..0..@.<..~..~.
02EE4134	00FCFFFF	FF030302	1B7ECEC0	0010	..~.....4A..
00030301	B4050402	0000E05E	2C831270	0020	p.,^.....

(...)

 TCPIPtrace XMT packet 5 at 1-JUL-1999 17:53:44.63

	IP Address	Port	Seq #	Ack #
Source	192.206.126.27	1023	876736002	4294966273
Destination	192.206.126.3	515		
		Packet Length		50
TCP flags	PSH ACK			
window	33580			

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
1B7ECEC0	4B80063C	0000BF80	32000045	0000	E..2....<..K..~.
01FCFFFF	02EE4134	0302FF03	037ECEC0	0010	..~.....4A.....
6E697270	666C6102	0000DFCD	2C831850	0020	P.,..... <u>alfprin</u>

Queue name.

0A74	0030	<u>t.</u>
------	------	-----------

(...)

 TCPIPtrace XMT packet 8 at 1-JUL-1999 17:53:44.81

	IP Address	Port	Seq #	Ack #
Source	192.206.126.27	1023	876736012	4294966274
Destination	192.206.126.3	515		
		Packet Length		68
TCP flags	PSH ACK			
window	33580			

			Hex	Count	Ascii
-----	-----	-----	-----	----	-----
1B7ECEC0	3780063C	0000C180	44000045	0000	E..D....<..7..~.
02FCFFFF	0CEE4134	0302FF03	037ECEC0	0010	..~.....4A.....
30416664	20303203	0000A135	2C831850	0020	P.,5....20 <u>dfA0</u>
2E6E6564	7572622E	3535326E	65643930	0030	<u>09den255.bruden.</u>

Spooled file name.

0A6D6F63	0040	<u>com.</u>
----------	------	-------------

(...)

```
-----
TCPIPtrace XMT packet 10 at 1-JUL-1999 17:53:44.83
      IP Address      Port      Seq #      Ack #

Source      192.206.126.27 1023 876736040 4294966275
Destination 192.206.126.3      515

      Packet Length      60

TCP flags      PSH ACK
      window      33580

      Hex      Count      Ascii
-----
1B7ECEC0 3E80063C 0000C280 3C000045 0000 E..<....<...>...~.
03FCFFFF 28EE4134 0302FF03 | 037ECEC0 0010 ..~.....4A.(....
206F7420 61746144 | 0000E74A 2C831850 0020 P.,J...Data to
      0A2E6465 746E6972 70206562 0030      be printed..

      File contents.
```

```
$ rsh/user="root" alfman "/usr/sbin/lpc stat"
```

```
alfprint:
```

```
printer is on device '/dev/tty00' speed 9600
```

```
queuing is enabled
```

```
printing is enabled
```

```
1 entry in spool area
```

```
Tue Jul 6 17:52:20 1999: Attempting to print dfA009den255.bruden.com
```

```
$
```


Printer Access Control

- The print server can grant or deny access:
TCPIP> SET SERVICE LPD/FLAGS=APPLICATION_PROXY
- The following commands cause LPD to authenticate remote users through the proxies:
TCPIP> ADD PROXY obrien /HOST=alfman /REMOTE_USER=obwon
TCPIP> ADD PROXY alf_users /HOST=alfman /REMOTE_USER="*"
- LPD can deny access from certain nodes:
TCPIP> SET SERVICE LPD /REJECT=HOSTS=(alfman,cuff)

TELNETSYM

- With TELNETSYM, you can set up and manage a remote printer attached to a remote terminal server or a network printer.
 - The printer functions as if it were directly connected to your system.
 - The Telnet symbiont functions like LATSYM for LAT software.
- The terminal server or network printer must support TCP/IP and Telnet. (Less expensive terminal servers support the LAT protocol only.)
- TELNETSYM passes print formatting functions, such as header and trailer page generation, pagination, queuing, and handling of multiple forms to the OpenVMS print symbiont.
- You must initialize the queue and specify TELNETSYM for the processor:

```
$ INITIALIZE /QUEUE /PROCESSOR=TCPIP$TELNETSYM -  
$_ /ON="printserver:port_on_printserver" TSYM_QUEUE
```

PC-NFS

- PC-NFS provides authentication and print services for personal computers.
 - PC client can associate a printer name with an OpenVMS print queue.
 - PC users must have an entry in the proxy database.
- Create and export a spool directory:

```
TCPIP> MAP "/PRINT_PC_SPOOL_DEV" DKA300:
TCPIP> ADD EXPORT "/PRINT_PC_SPOOL_DEV/SYS0/PCPRINT" /HOST=*
/OPTIONS=TYPELESS
```
- Define system logical names:

```
$ DEFINE /SYSTEM TCPIP$PCNFSD_SPOOLDEV DKA300:
$ DEFINE /SYSTEM TCPIP$PCNFSD_SPOOLEXPORT
"/PRINT_PC_SPOOL_DEV/SYS0/PCPRINT"
```


Introduction

This module presents options for troubleshooting network problems. The troubleshooting suggestions are broken into two major parts: configuration and network. Configuration troubleshooting involves checking to be sure that services are set up correctly. Network troubleshooting involves the more complex and dynamic world of networking software (and hardware) within and beyond your machine. Many logical names and log files pertinent to troubleshooting are introduced. Performance tuning suggestions using `sysconfig -r` are also examined.

Objectives

Upon completion of this module, you should be able to:

- Discuss troubleshooting options.
- Approach troubleshooting logically.
- Use service-specific log files.
- Use utilities to troubleshoot the network.
- Use `sysconfig -r` to alter network attributes.

Topics

- Troubleshooting Options
- Configuration Troubleshooting
- Network Troubleshooting
- Performance Tuning

Configuration Troubleshooting

- Configuration troubleshooting involves checking configuration displays and log files. You can configure the service (usually once) and perform the bulk of the troubleshooting during the configuration process or right after the configuration.

TCPIP> SHOW SERVICE TELNET

- A new service must include:
 - Name of the startup command file for the service.
 - Service port number.
 - Service process name.
 - User account information matching the information in the system user authorization file (SYSUAF.DAT).

- If the SHOW SERVICE display fails or is missing information, enter:

TCPIP> ANALYZE SERVICE

Network Troubleshooting

- Network troubleshooting requires a good conceptual knowledge of the network software and topology.
 - Issues arise throughout the life of the network.
 - Problems may be completely unanticipated.
 - Problems require common sense to solve.
- Problems could arise at any of the four layers of the protocol stack.
 - Detailed knowledge of the protocols is useful but not necessary.
 - Knowledge of the *big picture* of the network and a methodical approach is more valuable than voluminous details about protocols.

Common Sense Checks

- If only one application seems to be involved, focus your troubleshooting on that application.
Check the service configuration; it may be disabled or badly configured.
- If only one node reports the problem, focus on that node.
Check interfaces, services, and the routing table.
- If nodes within a particular subnet report the same problem, check the routing tables.

Troubleshooting Hints

- Never assume an answer. Keep considering all possibilities as you narrow your focus.
- Question the users carefully: what application, symptoms, dates, times, error messages, and so forth.
- Keep notes on what you have tried and the results.
- Always check the hardware: is everything attached, plugged in, lights on, and so forth.
- Quick checks on simple items may save you hours of effort.
- Use common and simple tools.
 - The ping, netstat, tcptrace, ripquery, arp, nslookup, and REPLY/ENABLE=NETWORK commands are an excellent collection of troubleshooting weapons.
 - If problems seem to be hardware oriented, invest in a cable tester.

BOOTP Troubleshooting

- View the client address for each incoming BOOTP request:
`$ DEFINE /SYSTEM TCPIP$BOOTP_TRACE 1`
- Request that log information be written to TCPIP\$BOOTP_STARTUP.LOG (found in SYS\$SYSDEVICE:[TCPIP\$BOOTP]):
`$ DEFINE /SYSTEM TCPIP$BOOTP_EXTLOG 1`
- Restart the service after creating logicals.

TFTP Troubleshooting

- Request logging of initial TFTP request from client in TCPIP\$TFTP_RUN.LOG (found in SYS\$SYSDEVICE:[TCPIP\$TFTP]):
`$ DEFINE /SYSTEM TCPIP$TFTP_EXTLOG n`

FTP Troubleshooting

- If FTP is not working, see if the remote FTP server is functioning. If the following command works, the problem is with the client; if it fails, the problem is with the server.
`$ TELNET ftp_server/PORT=21`
- If FTP is working but performing poorly, try altering the following logical names. (You can also add them to `SY$MANAGER:TCPIP$SERVICES_SETUP`).
 - Request larger window transfer size (send and receive) — default is 9216:
`$ DEFINE /SYSTEM /EXEC TCPIP$FTP_WNDSIZ 8192`
 - Request larger disk space allocation (blocks) when FTP creates a new file:
`$ DEFINE TCPIP$FTP_FILE_ALQ 700`
 - Request larger extension quantity when files grow:
`$ DEFINE TCPIP$FTP_FILE_DEQ 200`
 - Request that idle FTP sessions be timed out quicker:
`$ DEFINE TCPIP$FTPD_IDLETIMEOUT 600`

LPR/LPD Troubleshooting

- Request that log data be sent to TCPIP\$LPD_RCV_STARTUP.LOG:
\$ DEFINE /SYSTEM LPD_RCV 15 ! Logs extra inbound print messages.
- Request that log data be sent to TCPIP\$LPD_queueName.LOG:
\$ DEFINE /SYSTEM LPD_DEBUG 15 ! Logs extra outbound print messages.

NFS Troubleshooting

1. Enable NFS logging:
\$ TCPIP SET NFS_SERVER /ENABLE=ERROR
\$ TCPIP SET NFS_SERVER /ENABLE=OPCOM
2. Enable NFS tracing:
\$ TCPIP SET NFS_SERVER /DEBUG=180
3. Trigger the NFS problem.
4. Turn it all off:
\$ TCPIP SET NFS_SERVER /DISABLE=ERROR
\$ TCPIP SET NFS_SERVER /DISABLE=OPCOM
\$ TCPIP SET NFS_SERVER /DEBUG=0

Lower-Level Protocol Testing

The `ping` command tests lower levels of the protocol stack.

- If successful, the problem is probably further up the stack (above IP).
- If the `ping` fails and displays repeated messages (such as unknown host or network unreachable), the problem is probably on this (the sending) node.
- Try the `ping` with an address (rather than a name). If the address `ping` works, but the name `ping` fails, the problem is with DNS or the hosts file.
- If `ping` just seems to hang and displays no repeated messages, the problem is probably on some other node. It could be a routing table problem at the destination node or at an intermediate router.
- Try `tracert` or progressive pings (closest router, next closest, and so on) or `ping -R` (records the round-trip route taken by the packet).

Network Interface Problems

- The `ifconfig` command displays network interface information.
 - Make sure the interface is **up**. If it does not come up, check the interface hardware and the wires.
 - Make sure the interface is **running**. If not running, check the interface configuration and driver installation (if applicable).
 - Make sure the address, mask, and broadcast address are correct. Keep track of IP address topology as you add nodes to your network.
 - If `ping` reaches distant networks, but cannot reach local subnets, check the subnet mask.
 - Subnet masks are commonly defaulted or *inherited* and are usually only partially understood by administrators.

Connectivity

- If connectivity is intermittent, suspect duplicate IP addresses or hardware problems.
- The arp command displays the arp table, which relates IP addresses with MAC addresses.

Intermittent problems tend to be arp related. The arp table contains MAC addresses from the host that responded first (assuming two hosts with the same IP addresses).

- MAC addresses identify the equipment vendor in the first 3 bytes.
 - 08:00:2b — Compaq
 - 08:00:20 — SUN
 - 00:00:0c — Cisco



Find other MAC addresses listed at

<ftp://ftp.isi.edu.in-notes/iana/assignments/ethernet-numbers>

Network Traffic Problems

- The `netstat -i` command shows the count of packets queued (Queue column).
 - If the count is nonzero, check the cabling or interface hardware.
 - Many output errors (Oerrs) indicate physical connection problems or cable problems.
 - Many input errors (Ierrs) indicate network overload or hardware problems.
 - Collision greater than 5% of output packet count indicates that the traffic load needs to be divided.

Routing Problems

- Most network problems eventually come down to routing table problems.
 - The `netstat -nr` command yields valuable insights into routing issues.
 - Make sure data is reaching the wire. Check the packet use count column. Check to see if the data is going out an unexpected interface, or not going out at all.
 - If data is going out the correct interface, use the `TCPTTRACE` command to check if the data and addresses are correct. Check both source and destination addresses.
 - If adding a route to a subnet using a nondefault subnet mask, the mask must be part of the command syntax.
 - If adding a default route, you must exclude the `-netmask` syntax.

TCPIP> route add -net 192.206.126.96 192.206.126.12 -netmask 255.255.255.224

- `TCPTTRACE` yields basic TCP/IP protocol header information. Check the port numbers. Check the source and destination addresses. Is this node identifying itself correctly? Is the destination address reasonable?

Dynamic Routing

- Use TCPTRACE to check for routing updates.
- Check the routing table for correctness of additional routes (`netstat -rn`).
- Use ripquery to check for dynamic updates from routing information suppliers.

DNS Problems

- The key tool to DNS troubleshooting is `nslookup`.
- The public domain `dig` tool is also available (DNS Internet Groper).
- `nslookup` can eavesdrop on the server databases.
- If **TCPIP-E-BIND_NOSERVERS, default servers are not available** messages appear, check the database files for missing NS records. This typically happens after:

```
TCPIP> CONVERT/ULTRIX BIND /DOMAIN
```

Load Broker Troubleshooting

- Request that metric server diagnostics be placed in `TCPIP$BIND_METRIC_SERVER.LOG` (found in `SYSS$SPECIFIC:[TCPIP$BIND]`):

```
$ DEFINE/SYSTEM TCPIP$BIND_METRIC_DBG_LEVEL 2
```
- Request that diagnostics on BIND server load balancing be placed in `TCPIP$BIND_RUN.LOG` (found in `SYSS$SPECIFIC:[TCPIP$BIND]`):

```
$ DEFINE/SYSTEM TCPIP$BIND_CLUSTER_DBG_LEVEL 2
```
- You can view the metric rating of cluster members with `SYSS$COMMON:[SYSEXEC]TCPIP$METRICVIEW.EXE`.

Performance Tuning

- Partial connections.
 - Sockets are associated with a *listen queue*, which limits the number of pending connections.
 - The listen queue may fill with partial connections.
 - The solution may be to time out partial connections more quickly so the listen queue resource is freed up sooner:
TCP/IP> sysconfig -r inet tcp_keepinit=80
 - Default is 150. Unit is half-seconds.
- TCP keepalive.
 - This attribute of TCP is turned off by default. Connections may remain beyond their potential utility.
 - The solution may be to turn on tcp_keepalive_default, enabling dead connections to time out.
TCP/IP> sysconfig -r inet tcp_keepalive_default=1
- Delayed ACK algorithm.
 - TCP delays acknowledgement to try to minimize network traffic. Some applications benefit when there is no delay in the acknowledgement of received packets.
 - The solution may be to turn off delayed acknowledgements:
TCP/IP> sysconfig -r inet tcp_nodelack=1
TCP/IP> SET PROTOCOL TCP /NODELAY_ACK
- TCP retransmits too soon.
 - If TCP encounters a slow line, or delays in receiving acknowledgements, it may retransmit a packet sooner than desired.
 - The solution might be to increase the minimum TCP retransmit interval:
TCP/IP> sysconfig -r inet tcp_rexmit_interval_min=5
 - Default is 2. Unit is half-seconds.

- TCP socket listen queue is too small.
 - Incoming connect requests are getting dropped.
 - The solution might be to increase the maximum size of the listen queue:
TCPIP> sysconfig -r socket somaxconn=32767
- Application software specifies a listen queue limit that is too small.
 - You can alter the application limit without altering the source code:
TCPIP> sysconfig -r socket sominconn=32767
- Listen queue monitoring.
 - Monitor sobacklog_hiwat, sobacklog_drops, and somaxconn_drops through:
TCPIP> sysconfig -q socket
- Socket buffer sizes may be too big/small.
 - If buffer space is not available when needed, performance suffers.
 - View settings with:
TCPIP> SHOW PROTOCOL TCP /PARAMETERS (UDP also)
TCPIP> sysconfig -q inet
 - Alter settings with:
TCPIP> SET PROTOCOL TCP /QUOTA=SEND=65536 (UDP also)
TCPIP> sysconfig -r inet tcp_sendspace=65535 (UDP also)

Simple Mail Transfer Protocol (SMTP)

Module 15

Introduction

This module presents the Simple Mail Transport Protocol (SMTP). The SMTP acronym is sometimes used to encompass the SMTP, POP, and MIME protocols as well as any mail application that can handle the SMTP protocol. This module serves mainly as an overview, but it does show some of the protocols being used in a Telnet session directed to the SMTP port (25). POP and MIME support are also briefly discussed.

Objectives

Upon completion of this module, you should be able to:

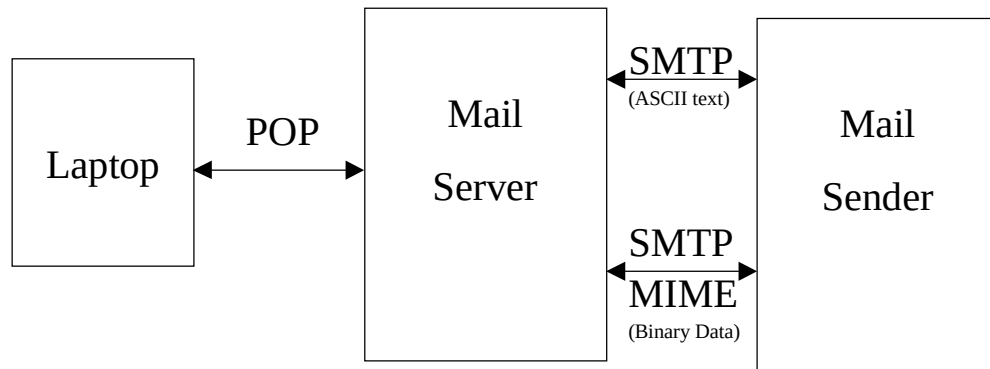
- Describe SMTP.
- Discuss SMTP as related to POP and MIME.
- Use Telnet to test SMTP.
- Use OpenVMS Mail to access SMTP.
- Start up and shut down SMTP and POP.

Topics

- SMTP Concepts
- Configuring SMTP
- Using Telnet to Test SMTP
- POP and MIME Concepts

SMTP Concepts

- Mail services for TCP/IP are typically built on three protocols:
 - Simple Mail Transport Protocol (SMTP)
 - Post Office Protocol (POP)
 - Multipurpose Internet Mail Extensions (MIME)
- SMTP is the backbone of all TCP/IP based email systems. It is used by mail applications that connect to TCP port 25.



SMTP, POP, and MIME Mail Protocols

SMTP Client Commands

Command	Description
HELO <sender>	Identifies the originating host.
MAIL FROM:<reverse path>	Identifies the sending user account.
RCPT TO:<forward path>	Address of the receiver, one for each recipient.
DATA	Begins a message.
QUIT	Ends the session.

Using SMTP Client Commands

\$ telnet 192.206.126.3 25

telnet to port 25 to eavesdrop on SMTP communications.

%TELNET-I-TRYING, Trying ... 192.206.126.3

%TELNET-I-SESSION, Session 01, host 192.206.126.3, port 25

-TELNET-I-ESCAPE, Escape character is ^]

220 alfman ESMTP Sendmail 8.8.8 (1.1.3.6) Thu, 8 Jul 1999 23:05:43 -0400 (EDT)

helo den255

250 alfman Hello den255 [192.206.126.27], pleased to meet you

HELO identifying the sending host.

mail from:<system@den255>

250 <system@den255>... Sender ok

MAIL FROM identifying the user.

rcpt to:<root@alfman>

250 <root@alfman>... Recipient ok

RCPT TO identifying the receiver.

data

354 Enter mail, end with "." on a line by itself

Here comes some junque mail.

.

250 XAA00000000821 Message accepted for delivery

DATA starts data transmission.

quit

QUIT ends the session.

221 alfman closing connection

%TELNET-S-REMCLOSED, Remote connection closed

-TELNET-I-SESSION, Session 01, host 192.206.126.3, port 25

\$

\$

SMTP Implementation

- TCP/IP Services for OpenVMS implements SMTP as an OpenVMS symbiont.
 - SMTP works with OpenVMS Mail.
 - The auxiliary server starts the SMTP receiver when a connect comes in to port 25.
- SMTP consists of two components: the sender (the queuing mechanism) and the receiver.
 - The receiver is activated by the auxiliary server.
 - The auxiliary server runs the command procedure specified in the SMTP service database entry.
 - It uses the TCPIP_SMTP account to run the receiver (server) program TCPIP\$SMTP_RECEIVER.EXE (found in SYS\$SYSTEM).

\$ mail

MAIL> send

To: SMTP%"obrien@alfman"

Note the special SMTP syntax (or obrien@192.206.126.3)

MAIL> send

To: alfman::obrien

Looks in DECnet databases.

%MAIL-E-LOGLINK, error creating network link to node ALFMAN

-SYSTEM-F-NOSUCHNODE, remote node is unknown

\$ tcpip show serv smtp/full

Service: SMTP

	State:	Enabled	
Port:	25	Protocol:	TCP
		Address:	0.0.0.0
Inactivity:	5	User_name:	TCPIP\$SMTP
		Process:	<u>TCPIP\$SMTP</u>
Limit:	10	Active:	0
		Peak:	0
File:	TCPIP\$SYSTEM:TCPIP\$SMTP_RECV_RUN.COM		
Flags:	Listen		
Socket Opts:	Rcheck Scheck		
Receive:	0	Send:	0
Log Opts:	Acpt Actv Dactv Conn Error Exit Logi Logo Mdfy Rjct Tim0 Addr		
File:	SYS\$SPECIFIC:[TCPIP\$SMTP]TCPIP\$SMTP_RECV_RUN.LOG		
Security			
Reject msg:	not defined		
Accept host:	0.0.0.0		
Accept netw:	0.0.0.0		

\$

- Start the sender:
\$ @SYS\$MANAGER:TCPIP\$SMTP_STARTUP
- Enable the receiver:
\$ TCPIP ENABLE SERVICE SMTP
- Stop SMTP:
\$ @SYS\$MANAGER:TCPIP\$SMTP_SHUTDOWN
- Incoming mail files are temporarily stored in
SYS\$SPECIFIC:[TCPIP\$SMTP]. File names:
 - 92022509472729_USERNAME.HOSTNAME
 - 92022509472729_SYSTEM.HOSTNAME

Post Office Protocol (POP)

- SMTP provides end-to-end delivery.
 - It functions as store-and-forward because DNS uses MX records that indicate mail servers for various zones.
- Most users in an SMTP environment eventually get their mail from a server.
 - The user's node moves mail from the server using a related protocol called POP.
- POP verifies user name and password, retrieves mail from the mail server, and then (typically) deletes it.
 - POP uses TCP port 110.
 - TCP/IP for OpenVMS uses POP v3 (which has a different set of commands than POP v2).
- The POP server is the mail repository, which:
 - Accepts and stores mail on your behalf, including when your client system is not connected.
 - Forwards mail to you at your request.
- PC clients and laptops use POP heavily.
- The POP server reads mail from the OpenVMS NEWMAIL folder.
 - If the TCPIP\$POP_LEAVE_IN_NEWMAIL logical is defined, mail is left in your NEWMAIL folder.
 - Otherwise, POP moves mail to the MAIL folder.
- To start and stop POP:
\$ @SYS\$MANAGER:TCPIP\$POP_STARTUP
\$ @SYS\$MANAGER:TCPIP\$POP_SHUTDOWN

Mail Addresses

- SMTP addresses are of the form *user@node.domain*.
- Mail exchange (MX) records direct mail to a particular address if a destination address is not available.
 - MX uses the information stored in MX records to route mail.
 - Each MX record contains a list of hosts that can accept mail for the specified destination.
 - The list is in order of routing preference — lower preference is tried first.
- You can create multiple MX records for the same destination domain. This provides:
 - Load balancing between mail routers.
 - Continuous routing of mail if one router is unavailable.
 - Routes for mail through a firewall.
- Use SET MX_RECORDS to add to the MX database:

```
TCP/IP> SET MX_RECORDS CUFF /GATEWAY=ALFMAN /PREFERENCE=100
TCP/IP> SET CONFIGURATION SMTP/GATEWAY=ALTERNATE=MIPMAN.BRUDEN.COM
TCP/IP> SET CONFIGURATION SMTP/ZONE=BRUDEN.COM
```
- Postmaster account receives all undeliverable mail.

Multipurpose Internet Mail Extensions (MIME)

- MIME is an extension to SMTP that allows users to send data that is more sophisticated. (SMTP was originally limited to 7-bit ASCII data.)
- MIME defines encoding techniques for carrying various types of data, and it creates extensions to the standard SMTP headers.
- For MIME mail to be decoded correctly:
 1. Configure the SMTP server with `/OPTION=TOP_HEADERS`.

The first lines of mail text after the four OpenVMS message header lines and the initial separating line must be the MIME headers.
 2. The POP server node must define the `TCPIP$POP_IGNORE_MAIL11_HEADERS` logical name.

Otherwise, MIME headers do not parse as message headers.
 3. The **From:** field must be recognized as an SMTP address.

Otherwise, the POP server sends the headers it creates from OpenVMS message headers as the headers of the mail message.
 4. Set `TCPIP$SMTP_JACKET_LOCAL` to 1 for all SMTP cluster systems.

This ensures that the mail will be delivered if the domain in the **From:** or **To:** fields appears local.
- If you see multiple header blocks, and the MIME version header is not in the first block, check steps 1 through 4.

Binary Mask Templates

Appendix A

Introduction

This appendix provides several templates for use with subnet mask problems and exercises. An example involving a Class B subnetting case is presented, as well as a brief look at supernetting and noncontiguous subnet masks.

Default Subnet Masks — Class B

- Given a Class B network address of 182.33, and using the default subnet mask of 255.255.0.0, you can define 2^{16} minus 2 (64k – 2 or 65,534) nodes within the Class B network.

- Class B network ID example: 182 .33
- Default Class B subnet mask: 255 .255 .0 .0
- In binary it is 11111111.11111111.00000000.00000000

- The boundary (line of demarcation) between the network bits and the node bits in an IP address is the point where the binary ones become zeros in the subnet mask.

Given a Class B IP address of 182.33.18.13 and a subnet mask of 255.255.0.0, what is the network number and what is the node number?

- The default case (no subnetting) is easy.

Draw the line of demarcation where the ones become zeros in the subnet mask.

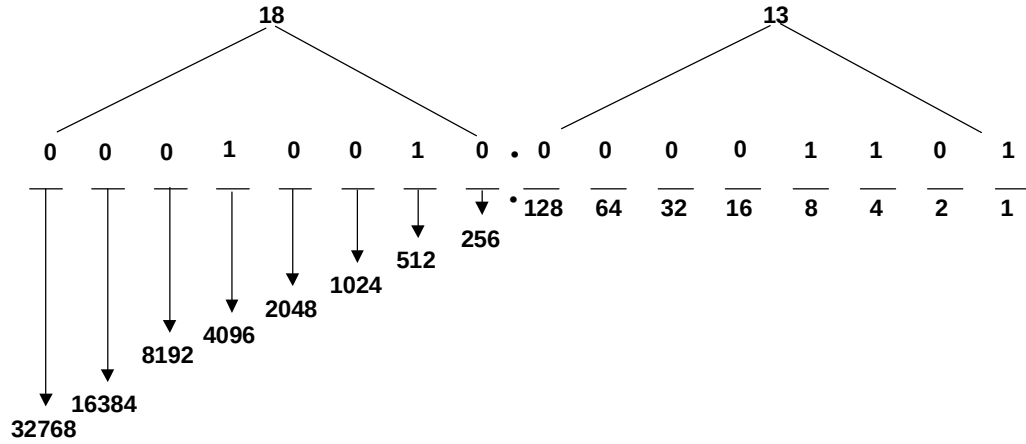
182	.33		.18	.13
11111111	.11111111		.00000000	.00000000

- The network number (182.33) is to the left of the line, and the node number (18.13) is to the right of the line.
- But what node number is 18.13?
 - Given this Class B example, note that you can use the rightmost 16 bits of the 32-bit address for node identification.
 - This provides 2^{16} minus 2 nodes maximum on the network (65,534).
 - The nodes can be numbered sequentially from 1 to 65,534.
 - This follows the rule of **not** using the broadcast addresses of all bits off (0) and all bits on (65,535)
- Again, what node number is 18.13?

Use the following templates to answer such questions.

Using Binary Bit Position Templates

- So for 18.13, represent the 18 as 00010010 binary and the 13 as 00001101 binary.



- The yield is $4096 + 512 + 8 + 4 + 1 = 4621$
- Node 18.13 is actually the 4,621st node of the possible 65,534 in the network.

Nondefault Subnet Masks — Class B Subnet Case

- Given:
 - Assigned Class B network ID is 135.100.
 - The company has 12 LANs, most with no more than 100 nodes on each.
 - However, several LANs have over 300 nodes each!
- What would be a reasonable subnet mask?
- Consider:
 - 255.255.128.0 (11111111.11111111.10000000.00000000)
Allows you to create two subnets (0 and 1). Each subnet can have 32,766 nodes ($2^{15} - 2$ or $32768 - 2$).
 - 255.255.192.0 (11111111.11111111.11000000.00000000)
Allows you to create four subnets (0, 1, 2, 3). Each subnet can have 16,382 nodes ($2^{14} - 2$ or $16,384 - 2$).
 - 255.255.224.0 (11111111.11111111.11100000.00000000)
Allows you to create eight subnets (0 through 7). Each subnet can have 8,190 nodes ($2^{13} - 2$ or $8,192 - 2$).
 - 255.255.240.0 (11111111.11111111.11110000.00000000)
Allows you to create 16 subnets (0 through 15). Each subnet can have 4,094 nodes ($2^{12} - 2$ or $4,096 - 2$).

- And:

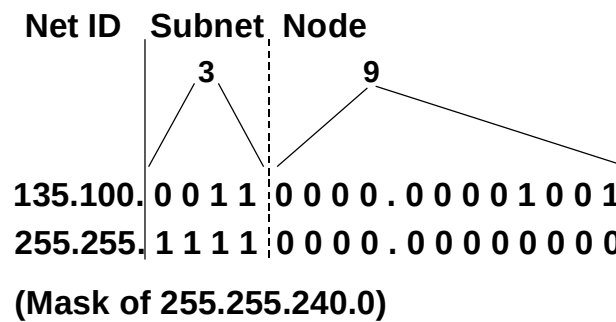
Subnet Mask	Number of Subnets	Nodes Each
255.255.248.0	32	2046
255.255.252.0	64	1022
255.255.254.0	128	510
255.255.255.0	256	254
255.255.255.128	512	126
255.255.255.192	1024	62
255.255.255.224	2048	30
255.255.255.240	4096	14
255.255.255.248	8192	6
255.255.255.252	16384	2

- Many Class B companies use the default Class C subnet mask (255.255.255.0) because of “ease of use” or “computer inertia.”
- They most likely do not need to set up 256 subnets, but it is easy to use and “it’s always been done this way.”
- This Class B case demands 12 subnets, so the choice is 255.255.240.0, which provides the 12 needed, plus an additional four for growth. (This choice also limits nodes to 4,094 maximum in each subnet.)
- Choose subnet numbers 1 through 12 for this case. What IP address would be used for node number 9 in subnet number 3?
- True subnetting breaks up the IP address into three components:
 - Network ID
 - Subnet number
 - Node number
- In this case, the network ID is **135.100**, the subnet number is **3**, and the node number is **9**.

Net ID	Subnet	Node
135.100.	---	---
255.255.	1 1 1 1	0 0 0 0 . 0 0 0 0 0 0 0 0

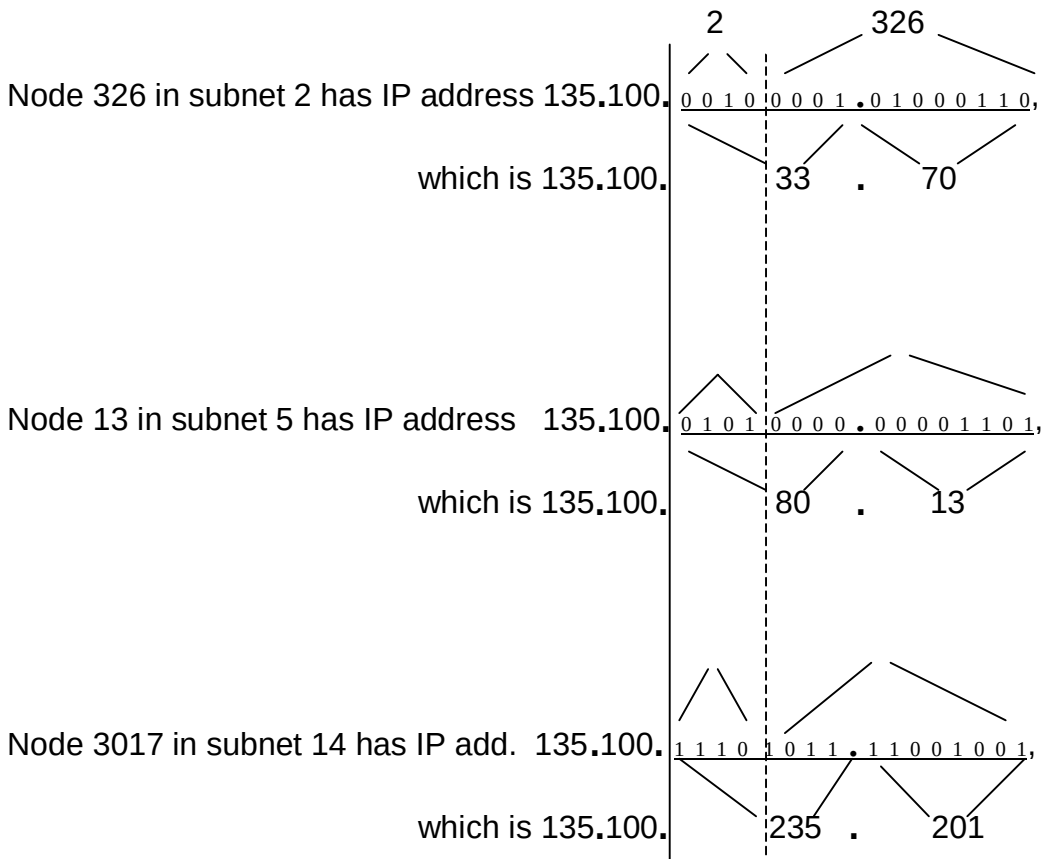
(Mask of 255.255.240.0)

- Using the lines to break up the 16 bits, represent a 3 in the subnet section (0011) and a 9 in the node section (0000.00001001).
- This yields 00110000.00001001 when the two binary values are concatenated into two 8-bit entities.



- Analyzing each series of 8 bits using the first template, note that 00110000 equals decimal 48 (32+16), and 00001001 equals decimal 9 (8+1).
- The final IP address is **135.100.48.9**.

- Furthermore, using the same analysis:



Broadcast Addresses – Class B Subnet Case

Broadcast addresses (all node bits on) for some of the Class B subnets would be:

Subnet #1 - 135.100. 0 0 0 1 1 1 1 1 . 1 1 1 1 1 1 1 1, 135.100.31.255

Subnet #5 - 135.100. 0 1 0 1 1 1 1 1 . 1 1 1 1 1 1 1 1, 135.100.95.255

Subnet #8 - 135.100. 1 0 0 0 1 1 1 1 . 1 1 1 1 1 1 1 1, 135.100.143.255

Subnet #11 -135.100. 1 0 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1, 135.100.191.255

Subnet #15 -135.100. 1 1 1 1 1 1 1 1 . 1 1 1 1 1 1 1 1, 135.100.255.255

Noncontiguous Subnet Masks

- Noncontiguous subnet masks are legitimate but usually not necessary.
- Consider the Class C network ID of 192.206.126 and a subnet mask of **255.255.255.160**.
 - 160 decimal is **10100000** binary.
 - So the mask has two extra bits set (beyond the default mask for Class C, 255.255.255.0).
 - Two subnet bits allows you to define up to four subnets (numbered 0 through 3).
 - Also, the remaining six node bits allow you to define up to 62 nodes in each subnet (numbered 1 through 62).
- Note that the analysis is more difficult with a noncontiguous subnet mask, but all of the previous concepts hold.
- A node with address 192.206.126.201 is actually on subnet 2 and is node 41 on that subnet.
 - 192.206.126.**11001001** (**Bolded** bits are subnet bits.)
 - 255.255.255.10100000 (Mask of 255.255.255.160.)So, the subnet bits are 1 0 (decimal 2), and the node bits are 1 0 1 0 0 1 (decimal 41).

Supernetting

- Suppose a large company needed a Class B IP address, but there were no more left!
 - A solution would be to give the company a series of contiguous Class C addresses to meet the company's addressing needs.
 - However, how do you inform the IP software that this series of Class C network IDs should be treated as one network?
- A subnet mask of 255.255.0.0 moves the line separating the network ID from the node number, 8 bit positions to the **left**.
- Therefore, ICANN gives the company 197.7.0.0 and a mask of 255.255.0.0, yielding the effect of a Class B address from several Class C addresses.
 - This process is referred to as *supernetting*.
- The use of the mask during routing decisions (instead of the address class) is called classless interdomain routing (CIDR).
 - CIDR requires that routing software distribute masks as well as addresses.
 - A routing table showing the following syntax indicates that the software can handle CIDR and supernetting.
197 . 7 . 126 . 42/16
 - The 16 indicates 16 bits of network ID information, followed by 16 bits of node information (same as 255.255.0.0).
192 . 206 . 126 . 45/24
 - The 24 indicates 24 bits of network ID information, followed by 8 bits of node information (same as 255.255.255.0).

NSLOOKUP Internal Commands

Appendix B

Introduction

This appendix provides a complete list of commands supported within the NSLOOKUP tool.

NSLOOKUP Internal Commands

LS

Lists the information available for the domain specified. The default output contains host names and their Internet addresses.

-a	Lists canonical names and aliases in the domain.
-d	Lists all entries in the domain.
-h	Lists CPU and operating system information for the domain, HINFO entries.
-m	Lists mail exchangers in the domain, MX entries.
-s	Lists well known services in the domain, WKS entries.
-t	Lists information of a specified type, the default is the default query type.
> file	Directs the output to the specified file
>> file	Appends the output to the specified file

LSERVER

Changes the default server to the specified server. If necessary, this command uses the initial default server to translate the specified server name into an address.

SERVER

Changes the default server to the specified server. If necessary, this command uses the current default server to translate the specified server name into an address.

FINGER

Finger one or all users on a host. A host is the last host for which information was requested. If a name is specified, detailed information is displayed for that user. If no name is specified, a list of logged on users is displayed.

ROOT

Changes the server for the root of the domain name space. The host **ns.nic.ddn.mil** is used.

SET

Changes the options that affect lookup operations. The current options are displayed with `set all` or `set ALL`.

SET Options

all, debug, d2, defname, recurse, vc, domain, search, srchlist, root, retry, timeout, querytype, type, class, ignoretc

all	Displays the current values of the nslookup options and information about the current default server. SET ALL also prints the server state information.
debug	Turns on debugging mode. Verbose information is printed about the packet sent to the server and the resulting answer.
d2	Turns on exhaustive debugging mode.
defname	Appends the default domain name to every lookup.
recurse	Tells the BIND/Hesiod server to query other servers if it does not have the information.
vc	Uses a TCP connection when sending requests to the server.
domain	Changes the default domain to the domain name specified. The default domain name is appended to all lookup requests if the defname option is set.
search	If a lookup request contains at least one period but does not end with a trailing period, appends the domain names in the domain search list until an answer is received.
srchlist	Changes the set of domains that are searched if a name is not found in the default domain. The default search list is the domain of the current node, and up to two successive parent domains up to the root.
root	Changes the name of the root server to the host name specified. This affects the root command.
retry	Sets the number of retries to the number specified. If a reply to a request is not received within a certain amount of time (changed with set timeout), the request is resent. The retry value controls how many times a request is to be resent before giving up.
timeout	Changes the time-out interval for waiting for a reply to the number specified (in seconds).
querytype	Changes the type of information returned from a query. Allowed values:
	ANY Matches any type
	A Internet address
	AAAA IPv6 Internet address
	CNAME Canonical name for an alias
	HINFO Host information
	MX Mail routing information
	PX X.400 mail mapping
	NAPTR Naming authority pointer
	NS Authoritative name server
	NSAP NSAP address
	PTR Domain name pointer
	SOA Start of authority zone
	SRV Server selection
	TXT Text strings
	WKS A well-known service
	Default: set querytype=A
	type A synonym for querytype
	class Changes the class of information returned from a query. Allowed values:
	IN Internet
	HS Hesiod
	any any
	Y Y
	Default: set class=IN
ignoretc	Ignore packet truncation errors.

Introduction

This appendix describes the dynamic routing protocols in more detail. Focus is on RIP and OSPF, but IS-IS, IGRP, and EIGRP are also examined.

Routing Information Protocol (RIP)

RIP is a distance-vector protocol.

- Decisions about which path to use are based on distance.
- Distance is represented solely by hops.
- Simple implementation is based on routed daemon.
- Complexities of today's networks strains the capabilities of RIP.
- A user might choose a slower or more expensive route because it has fewer hops.

RIP Dynamic Routing

- With dynamic routing through RIP, each router broadcasts its routing table every 30 seconds onto each interface, which:
 - Allows adjacent routers to “learn” everything their neighbor knows.
 - Adds new information to the routing table.
 - Replaces any less efficient routes with more efficient routes.
 - Each router adds 1 to the hops metric as it learns new routes from its neighbors.
 - Eventually all routers arrive at a stable view of the network.
- Hops value is between 1 and 15. A value of 16 indicates the network is unreachable.
- RIP messages can contain up to 25 network routes.
- RIP uses UDP port 520.

RIP Convergence

Convergence time is the time it takes for a change in routing topology to be propagated throughout the Internet.

- Convergence takes place one hop at a time (every 30 seconds).
- Therefore, a 15-hop network could take up to seven minutes for a change to be reflected to the other side of the environment (worst case).
- During convergence, the network is in a state of flux.
- Some routers know one thing; others know something else.
- This is why RIP limits networks to 15 hops.
- Extended RIP (ERIP) allows the network diameter to increase to 128.
- During the convergence window, the network can be unstable (could have routers with routing tables that point to each other).

RIP Count to Infinity

- Router A sends out information on network 114.124, but then the interface to 114 fails. Router A marks the routing table information for network 114 as metric 16. Within 30 seconds, Router B sends out its update message, which contains information about finding network 114 at a cost of 2 (which is far better than 16). (A is not aware that B intends to use A to get to network 114.) So now, A's table points to B to get to 114, and B's table points to A.
- Furthermore, when Router A sends out its 30-second update, B notices that the hops cost to get to 114 has increased. Because A originally told B about 114, B is compelled to reflect the increase in its table. This goes on continuously, with each router telling the other about an increase that must be reflected in the routing table. Each time, the cost is bumped by 1.
- Eventually the cost will be calculated at 16, and the routers will figure out that the cost is infinite.
- This process could leave the network unstable for up to four minutes.

RIP Count to Infinity Solutions

- Poison reverse solution:
 - Set up the routing update message for a particular interface so that a cost of 16 is associated with any network learned about from that interface.
 - In this solution, Router B avoids telling Router A how to get to networks that B originally learned from A.
- Split horizon solution:
 - This solution states that routing information should not be sent backward.
 - Router A would never learn that B could get to its networks.

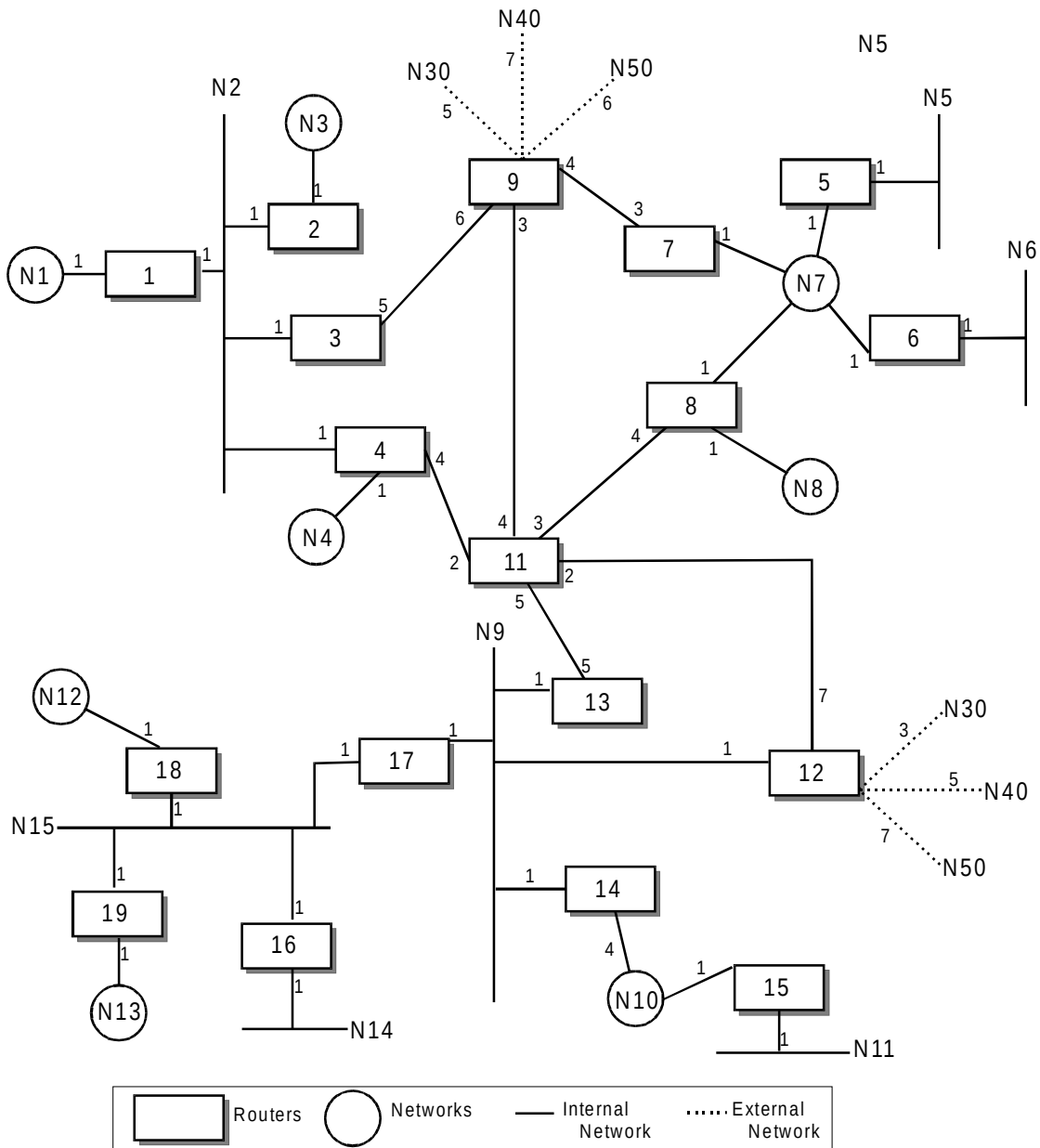
RIP Limitations

- Limited network diameter.
 - Maximum of 15 routers between two networks.
 - ERIP allows 128.
 - Choice of routing over five Ethernets instead of one 9.6-kbps, point-to-point link would be the 9.6!
 - You can artificially set the hop count metric, but this further limits your network diameter.
- Convergence is the time it takes to learn about network changes.
 - Routers will have erroneous information for some time.
 - Routers may try to send data to a failed network.
- Routing table entry aging.
 - Entries can remain for up to 180 seconds without a refresher update.
 - If a router fails, others will still think it is available for up to three minutes.
- Subnetting.
 - RIP assumes the same subnet mask throughout the RIP environment. (RIP2 passes the subnet mask as part of the routing update).
 - Multiple masks could create “black holes” (portion of the network not seeable by the rest of the network).

Open Shortest Path First (OSPF)

- OSPF was created specifically for IP as a replacement for RIP.
- OSPF is a link-state, shortest-path-first protocol.
 - Link-state tracks the operational state of every interface.
 - Changes to the state trigger sending of routing update messages.
- OSPF reduces the routing update messages.
 - Only the changed information is sent.
 - CPU and memory costs are higher.
- OSPF uses Dijkstra's algorithm to build the routing environment:
 - Computes all the ways to get to each destination.
 - Creates a topological database with redundant paths.
 - Topological database is a data structure allowing for much faster convergence. (< 20 secs.)
- Shortest-path-first means that each router has its own view of the internetwork.
 - Path is the steps needed to get from one point to another.
 - Uses a configurable, path-cost variable.
 - Shortest path tree shows the one best path to each destination in the internetwork along with the total path cost.

OSPF Routing



OSPF Areas

An OSPF area is a logical collection of networks grouped together and connected by routers.

- Within an area, all routers disclose complete information about the known networks and maintain identical topological databases.
- Summary information is passed to other areas, which:
 - Speeds up the convergence calculations.
 - Saves space in routing tables.
 - Requires a well-planned addressing scheme.
- An OSPF area is identified by a 32-bit area ID.
 - ID 0.0.0.0 is the backbone area.
 - The backbone area contains network and routers that connect to other areas.

OSPF Router Classifications

- Internal router
 - Interface connected to networks in the same area.
 - Maintains information about all networks in the area.
 - Gets summary information about networks in other areas.
- Area border router
 - Has network connections to more than one area.
 - Maintains full information about its own area.
 - Maintains full information about the backbone area.
 - Gets summary information about other areas.
 - Sends summary information to other areas.
- Backbone router
 - Has connections to the backbone.
 - Maintains information about the backbone.
 - Distributes summarized area information.
- Autonomous system boundary router
 - Attached to networks outside the autonomous system.
 - Receives and distributes information about external networks.

OSPF Variable-Length Subnetting

- Big win for OSPF!
- The subnet mask is part of the routing message.
- You can choose any subnet mask for a particular network.
- You can save many addresses that might be wasted in RIP.
- Variable-length subnetting also supports unnumbered point-to-point networks.

OSPF Flooding

- Flooding describes how OSPF information is sent throughout the environment.
- It propagates information from router to router.
- Each router sends the information to its connected networks.
- Flooding occurs only when a change has been sensed.
- Flooding ensures the receipt of updates, and it expects acknowledgements.

IS-IS

- Intermediate system to intermediate system
 - OSI parlance for router.
 - Potentially more portable than OSPF.
 - Based on a link-state strategy.
- Dual IS-IS:
 - Potentially supports more than just IP routes (CLNP).
 - OSI specification supports just CLNP (Connectionless Network Protocol).
- Separates routing domain into hierarchical areas
 - Each area contains one or more networks.
 - IS-IS areas are stub areas with only one exit.
 - No subnet identifiers.
 - Hosts identify themselves to routers through ES-IS protocol.
- Contains subprotocols
 - Hello discovers neighbors and elects designated routers.
 - Flooding protocol distributes link-state records.

IGRP and EIGRP

- Users could either wait for the IETF to come up with OSPF or design their own solution to the limitations of RIP.
- The Cisco Company responded by creating IGRP (1988).
- IGRP uses the distance-vector strategy, broadcasting updates (like RIP) every 90 seconds.
- Provides:
 - Composite metrics including delay, bandwidth, reliability, and load. (Formula includes portions of the multiple metrics.)
 - Protection against loops through one of the following:
 - ◆ Split horizon and route poisoning (as the hop count increases, treats the path as unusable for 180 seconds).
 - ◆ As soon as the update is confirmed (90 seconds).
 - Multipath routing and better default route handling.
- Extended IGRP:
 - Cisco engineers implemented a version of OSPF but continued development of IGRP. Many felt that distance-vector technology provides more flexibility with less complexity.
 - Incorporates sophisticated extensions to the distance vector protocol.
 - ◆ Maintains multiple information for each destination.
 - ◆ Uses a diffusing update algorithm (DUAL) to eliminate transient loops.
 - Uses an acknowledgement technique with five message types: Hello, Update, Query, Reply, and Request.

Introduction

This appendix is a reference for definitions of wire-based networking terminology. Some software terms are discussed as well.

Network Fundamentals

LAN cables can be:

- Unshielded Twisted Pair (UTP)
 - Already exists in most buildings (cheap)
 - RS-232, 10Base T, 100Base T
- Shielded Twisted Pair (STP)
 - For longer distances and larger, faster LANs
 - Commonly used for Ethernet (10Base T) and Token Ring networks
- Coaxial cable
 - ThinNet using an internal transceiver (on card)
 - ThickNet requiring an external transceiver
 - Good for longer distance, fast LANs
 - Fairly immune to electrical interference
- Fiber optic cable
 - Data transferred by light rather than electricity
 - Longer cable runs; less noise susceptibility; excellent capacity
 - Higher cost; more installation expertise needed
 - Fiber Distributed Data Interface (FDDI) 100Mbps

Error Detection

- Parity checking is an asynchronous error checking technique.
 - Asynchronous transfer is most commonly used in dial-ups.
 - Parity bit is just before the stop bit in a byte transfer.
 - Bit 0 — start bit (positive voltage) to get the receiver's attention
 - Bits 1–8 — byte to be transferred
 - Bit 9 — parity bit
 - Bit 10 — stop bit
- Message 1 1 1 1 0 0 0 1
 - Oops! 1 0 1 0 0 0 0 1
- So, parity is not too accurate for most forms of data transmission.
- Synchronous transfer frames the data with control bytes fore and aft.
 - ♦ Control information is usually addresses and indirect commands to the receiver.
 - ♦ Bytes are not individually framed.
- Cyclic redundancy check is used by most framed synchronous data transfer applications.
 - Also known as frame check sequence and block check characters.
 - CRC 32 is used for LANs.
 - Use it to check an entire frame of data (not 1 byte).
 - The data is divided by a constant number and the remainder of the division is sent as a validation check.
- Recovery is usually achieved by some form of retransmission.

Network Topologies

- Star (or hub) topologies are common.
 - Can have management or troubleshooting software built in.
 - Good for closets and small spaces.
 - Terminal servers use this topology.
 - Easy cable changes; centralized management; easy to expand.
 - Single point of failure; more cable needed; hubs can be expensive.
- Bus topology is used primarily by Ethernet.
 - Simple cabling, high reliability.
 - Fault diagnosis is painful; bus can be a bottleneck.
- Ring topology.
 - Sorter cables are less expensive.
 - No hub is less expensive.
 - A single node can be a point of failure.
 - Changes disrupt the LAN.
- LANs typically use a digital encoding scheme to describe serial binary digits.
 - Manchester for Ethernet (high–low = 0; low–high = 1).
 - Differential Manchester for Token Ring (change signifies switch from 0 to 1).

- IEEE specifications include:
 - 802.3 — CSMA/CD (used by Ethernet)
 - 802.4 — Token Passing Bus
 - 802.5 — Token Passing Ring
 - 802.8 — Fiber Optics
 - 802.10 — Security
- Media access control techniques are required to handle many users accessing the same wire at the same time.
 - CSMA/CD (carrier sense, multiple access with collision detect)
 - ♦ Nondeterministic (free for all).
 - ♦ 10 Mbps.
 - ♦ While one node transmits, any others wishing to transmit can sense that the carrier is busy.
 - ♦ Two machines can sense a noncarrier at the same time, resulting in a collision.
 - ♦ All transmitters must back off for a random amount of time and then retransmit.
 - ♦ Minimum frame size is needed to allow the transmitter to sense the collision (64 bytes).
 - Token passing (bus or ring)
 - ♦ Deterministic (wait your turn).

Cable Naming Conventions

- 10Base2 — 10 Mbps; baseband ThinNet coax; 185 meters maximum
- 10Base5 — 10Mbps; baseband ThickNet coax; 500 meters maximum
- 10BaseT — 10 Mbps; baseband unshielded Twisted Pair
- 10BaseFL — 10 Mbps; baseband fiber optic links
- 100BaseVG — 100Mbps; baseband category 3 Twisted Pair
- 100BaseT — 100Mbps; baseband category 5 Twisted Pair
- ThinNet
 - Coaxial not longer than 185 meters.
 - 50-ohm terminator at each end.
 - .5 meter intervals between stations.
 - Maximum 30 stations on a single segment.
 - Transceiver on card.
- ThickNet
 - 10Base5 has a separate transceiver.
 - Maximum 500 meters per segment.
 - 50-ohm terminator at each end.
 - 2.5 meters between stations.
 - Transceiver cable can be up to 50 meters.
 - Maximum 100 stations per segment.

Hardware Technologies

- Ethernet is a packet-switched LAN technology standardized in 1978.
 - IEEE version of the standard is 802.3.
 - Original Ethernet cable was coaxial 0.5 inch in diameter and up to 500 meters long.
 - ♦ Connections were made by a transceiver device.
 - ♦ Host adapter connects into the bus on the computer.
 - ♦ Cable and transceiver were expensive.
 - ♦ Cable was difficult to install.
 - ThinWire Ethernet is a thinner, less robust, more flexible coaxial cable.
 - ♦ Less expensive.
 - ♦ The transceiver is replaced with direct connection from the computer to Ethernet through high-speed digital circuits.
 - Twisted Pair Ethernet is noncoaxial cable (like phone wire).
 - ♦ Even less expensive.
 - ♦ Known as 10Base-T.
 - ♦ Connects each computer to an Ethernet hub device that simulates signals from an Ethernet cable.
 - ♦ Connection must be less than 100 meters.
 - ♦ Connection appears as a transceiver to the computer.
- Connections.
 - ThickWire requires an attachment unit interface (AUI) connector.
 - ThinWire requires a BNC connector.
 - Twisted Pair requires an RJ45 connector (looks like a modular telephone hookup).

FDDI

- FDDI stands for Fiber Distributed Data Interface, an optical transport.
 - FDDI uses glass fibers and transfers data by encoding it in pulses of light.
 - It is not harmed by electrical noise.
- FDDI is another standard for the low levels of the protocol stack.
- It runs 10 times faster than the Ethernet's 10Mb/second (100Mb/second).
- It is secure because no signals are generated as with an electrical medium.
- An FDDI network could consist of dual-counter rotating rings with up to 1,000 workstations over 125 miles (Token Ring technology).
- Dual rings allow for quick healing of network breaks.
- Is a Token Ring network to ensure fair and orderly access.

Token Ring

- A station wanting to transmit must wait for the token.
 - Adds a data frame on the end of the token.
 - Adds source and destination addresses.
 - Destination machine picks up the message but lets it continue on the ring.
 - Source machine removes the message frame and releases the token.
- Token format:
 - Byte 1 — start delimiter
 - Byte 2 — access control information
 - ♦ Bits 0–2, priority (1=high; 0=normal)
 - ♦ Bit 3, token bit (1=frame; 0=token)
 - ♦ Bit 4, monitor control
 - ♦ Bits 5–7, monitor count or reservation priority
 - Byte 3 — end delimiter
 - ♦ Bit 6, intermediate bit (0=single or last; 1=intermediate)
- Data frame format:
 - Byte 1 — start delimiter
 - Byte 2 — access control
 - Byte 3 — frame control
 - ♦ Bits 0–1 (00= ring control data)
 - ♦ (01= user data)
 - Bytes 4–15 — destination and source addresses (6 bytes each)
 - Bytes 16–21 — route information (ring and bridge numbers)
 - Bytes 22–22+<17.8K — data
 - Next 6 bytes are status and error control information

FDDI Frames

- FDDI frames are measured in 4-bit units called symbols.
- Maximum length is 4000 symbols (4500 bytes).
- Good for large volume transfers.
- Not exactly the same as 802.5 specification.

Wide Area Networks

- Wide area networks (WANs) usually involve longer distances.
- WAN-specific interfaces such as RS-232 define the physical interface and electrical signaling used.
 - RS-232
 - ◆ Slow, short cable runs
 - ◆ Very common
 - ◆ Up to 28.8 Kbps (two-wire dial-up)
 - ◆ Standard 25-pin connector
 - ◆ Serial interface
 - RS-449
 - ◆ Fast, long cable runs
 - ◆ Developed in 1977 by Electronics Industry Association (EIA)
 - ◆ Serial interface
 - ◆ Supposed to replace RS-232
 - ◆ Has a 37-pin connector
 - ◆ Several additional status, service, and test functions beyond RS-232
 - V.35
 - ◆ Medium speed, long cable runs
 - ◆ Defined by CCITT
 - ◆ Has a 34-pin, Winchester-style connection
 - ◆ Used by the phone company for digital data service (DDS)
- WAN uses WAN-specific technologies such as packet switching.
- Signaling is between data communications equipment (DCE — such as modems), and data terminating equipment (DTE — such as computers, terminals, and printers).

Asynchronous Transfer Mode (ATM)

Asynchronous transfer mode (ATM):

- Can switch data at gigabit speeds.
- Requires expensive equipment.
 - High-speed switches connecting to computers and other switches.
 - Optical fibers for connections.
- Uses fixed-sized frames called cells (53 bytes long).
 - With 5 bytes for header information.
- Is connection oriented (not packet switched).
 - Host must interact with switch before sending.
- Is designed to integrate voice, video, and data applications over the same network.
- Has three components at the physical level:
 - ATM adaptation layer, for converging networks into ATM.
 - ATM switching, for virtual path and virtual connection information. VP connects the machines, and VC connects the users.
 - Physical layer, for the physical transmission facilities.
- The ATM cell contains:
 - Bytes 0–4 — cell header
 - ♦ 4-bit flow control
 - ♦ 8 bits VP ID
 - ♦ 16 bits VC ID
 - ♦ 3 bits data type (user data or ATM management data)
 - ♦ 1-bit cell loss priority
 - ♦ 8 bits header error correction
 - Bytes 5–52 — payload

WAN Protocols

- Link layer protocols for WANs include Serial Line Interface Protocol (SLIP) and Point-to-Point Protocol (PPP).
- PPP allows you to pass a LAN protocol packet over a WAN PPP link. The header contains a PPP control field (2 bytes) (0021=TCP/IP).

