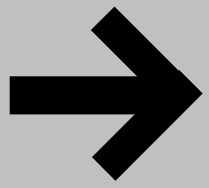
student guide

hp-ux system and network administration II

H3065S C.03



training

© 2003 Hewlett-Packard Development Company, L.P.

OSF, OSF1, OSF/Motif, Motif, and Open Software Foundation are trademarks of the Open Software Foundation in the U.S. and other countries.

UNIX® is a registered trademark of The Open Group.

X/Open is a trademark of X/Open Company Limited in the UK and other countries.

All other product names mentioned herein may be trademarks of their respective companies.

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

H3065S C.03
Student Guide
04/03

Contents

Overview
Course Description 1
Student Performance Objectives 1
Student Profile and Prerequisites 7
Curriculum Path 7

Module 1 — LAN Concepts
1–1. SLIDE: What Is a Network? 1-2
1–2. SLIDE: The OSI Model in a Nutshell 1-4
1–3. TEXT PAGE: OSI Worksheet 1-6
1–4. SLIDE: Media Access Control (MAC) Addresses 1-7
1–5. SLIDE: Internet Protocol (IP) Addresses 1-9
1–6. SLIDE: IP Network Classes 1-12
1–7. SLIDE: The IP Netmask 1-15
1–8. SLIDE: The IP Network Address 1-17
1–9. SLIDE: The IP Broadcast Address 1-19
1–10. SLIDE: The IP Loopback Address 1-21
1–11. SLIDE: Obtaining an IP Address 1-22
1–12. SLIDE: IP Address Examples 1-25
1–13. SLIDE: Host Names 1-26
1–14. SLIDE: Converting IP Addresses to MAC Addresses 1-28
1–15. SLIDE: Populating the ARP Cache 1-30
1–16. SLIDE: Putting It All Together 1-32
1–17. SLIDE: Managing Packet Flow with TCP 1-33
1–18. SLIDE: Managing Packet Flow with UDP 1-35
1–19. SLIDE: Sending Data to Applications via Ports 1-37
1–20. SLIDE: Managing Ports with Sockets 1-39
1–21. SLIDE: More on Socket Connections 1-41
1–22. SLIDE: Revisiting the OSI Model 1-43
1–23. REVIEW QUESTIONS: LAN Concepts and Components 1-44

Module 2 — LAN Hardware Overview
2–1. SLIDE: LAN Hardware Components 2-2
2–2. TEXT PAGE: OSI Worksheet 2-4
2–3. SLIDE: Transmission Media 2-5
2–4. SLIDE: LAN Topologies 2-9
2–5. SLIDE: LAN Access Methods 2-11
2–6. SLIDE: Ethernet 802.3 Interface Cards 2-13
2–7. SLIDE: Token Ring 802.5 Interface Card 2-18
2–8. SLIDE: FDDI Ring Interface Cards 2-20
2–9. SLIDE: Repeaters 2-22
2–10. SLIDE: Hubs 2-23
2–11. SLIDE: Bridges 2-24
2–12. SLIDE: Switches 2-26
2–13. SLIDE: Routers and Gateways 2-28
2–14. SLIDE: Firewalls 2-30
2–15. SLIDE: Pulling It All Together 2-31

© 2003 Hewlett-Packard Development Company, L.P.

Contents

Module 3 — Configuring IP Connectivity
3–1. SLIDE: TCP/IP Configuration Overview....3-2
3–2. SLIDE: Installing LAN Software....3-4
3–3. SLIDE: Checking LANIC Autoconfiguration....3-6
3–4. SLIDE: HP-UX Network Startup Files....3-8
3–5. SLIDE: Configuring Link Layer Connectivity....3-9
3–6. SLIDE: Configuring IP Connectivity....3-12
3–7. SLIDE: Configuring IP Multiplexing....3-15
3–8. SLIDE: Configuring **`/etc/hosts`**....3-17
3–9. LAB: Configuring Network Connectivity....3-19

Module 4 — Configuring IP Routing
4–1. SLIDE: Routing Concepts....4-2
4–2. SLIDE: Routing Tables....4-3
4–3. SLIDE: Viewing Routing Tables....4-5
4–4. SLIDE: Configuring Static Routes....4-7
4–5. SLIDE: Configuring a Default Route....4-9
4–6. SLIDE: Configuring Routes in **`/etc/rc.config.d/netconf`**....4-11
4–7. LAB: Configuring Routing....4-12

Module 5 — Configuring Subnetting
5–1. SLIDE: Limitations of Large Networks....5-2
5–2. SLIDE: Subnetting Concept....5-4
5–3. SLIDE: IP Addresses in a Subnetted Network....5-6
5–4. SLIDE: Netmasks in a Subnetted Network....5-7
5–5. SLIDE: Subnet Addresses....5-9
5–6. SLIDE: Host IP Addresses on a Subnet....5-11
5–7. SLIDE: Limitations of Subnetting on an Octet Boundary....5-13
5–8. SLIDE: Subnetting on a Non-Octet Boundary....5-14
5–9. TEXT PAGE: More Subnetting on a Non-Octet Boundary....5-16
5–10. SLIDE: Routers in a Subnetted Network....5-17
5–11. SLIDE: Configuring Subnetting....5-18
5–12. TEXT PAGE: Class B and Class C Subnetting Reference Sheet....5-20
5–13. LAB: Configuring Subnets....5-21

Module 6 — Troubleshooting Network Connectivity
6–1. SLIDE: Network Troubleshooting Tools Overview....6-2
6–2. SLIDE: Potential Network Connectivity Problems....6-3
6–3. SLIDE: The **`lanscan`** Command....6-5
6–4. SLIDE: The **`linkloop`** Command....6-7
6–5. SLIDE: The **`lanadmin`** Command....6-9
6–6. SLIDE: Example **`lanadmin`**....6-11
6–7. SLIDE: The **`arp`** Command....6-14
6–8. SLIDE: The **`ping`** Command....6-16
6–9. SLIDE: The **`netstat -i`** Command....6-18
6–10. SLIDE: The **`netstat -r`** Command....6-20
6–11. SLIDE: The **`nslookup`** Command....6-22
6–12. LAB: Troubleshooting Network Connectivity....6-24

© 2003 Hewlett-Packard Development Company, L.P.

Module 7 — Starting Network Services
7–1. SLIDE: Starting System and Network Services 7-2
7–2. SLIDE: Run Levels 7-4
7–3. SLIDE: **/sbin/rc*.d** Directories 7-7
7–4. SLIDE: S/K Script Naming Convention 7-9
7–5. SLIDE: **/sbin/init.d/*** Scripts 7-11
7–6. SLIDE: What's in an **init.d** Script? 7-12
7–7. SLIDE: **/etc/rc.config.d/*** Files 7-14
7–8. SLIDE: Pulling It All Together 7-16
7–9. SLIDE: Viewing Console Messages When Changing Run Levels 7-18
7–10. SLIDE: Creating Custom Start Scripts 7-20
7–11. LAB: Starting Network Services 7-24

Module 8 — NFS Concepts
8–1. SLIDE: What Is NFS? 8-2
8–2. SLIDE: What Files Should I Share via NFS ? 8-4
8–3. SLIDE: NFS Servers and Clients 8-6
8–4. SLIDE: The NFS Remote Procedure Calls 8-8
8–5. SLIDE: The **portmap** and **rpcbind** Daemons 8-10
8–6. SLIDE: NFS Stateless Servers 8-12
8–7. SLIDE: NFS PV2 versus NFS PV3 8-14
8–8. SLIDE: NFS versus CIFS 8-16

Module 9 — Configuring NFS
9–1. SLIDE: NFS Configuration Considerations 9-2
9–2. SLIDE: Configuring NFS Servers and Clients 9-4
9–3. SLIDE: Keep UIDs and GIDs Consistent 9-5
9–4. SLIDE: Ensure that the NFS Subsystem Is in the Kernel 9-8
9–5. SLIDE: Edit NFS Server's Configuration File 9-9
9–6. SLIDE: Start NFS Server Daemons 9-12
9–7. SLIDE: Create the **/etc/exports** File 9-14
9–8. SLIDE: Export the Directories 9-18
9–9. SLIDE: Check the Server Configuration 9-20
9–10. SLIDE: Ensure that the NFS Subsystem Is in the Kernel 9-22
9–11. SLIDE: Edit the Client's Configuration File 9-23
9–12. SLIDE: Start NFS Client Daemons 9-25
9–13. SLIDE: Create a New Entry in **/etc/fstab** 9-27
9–14. SLIDE: Mount the NFS File System 9-29
9–15. SLIDE: Check the Client Configuration 9-33
9–16. SLIDE: Review: Configuring NFS Servers and Clients 9-35
9–17. SLIDE: Common NFS Problems 9-36
9–18. SLIDE: Monitoring NFS Activity with **nfsstat** 9-38
9–19. LAB: Configuring NFS 9-40

Module 10 — Configuring AutoFS
10–1. SLIDE: AutoFS Concepts 10-2
10–2. SLIDE: AutoFS Maps 10-4
10–3. SLIDE: AutoFS Commands and Daemons 10-6
10–4. SLIDE: Starting and Stopping AutoFS 10-8
10–5. SLIDE: Configuring the AutoFS Master Map 10-11
10–6. SLIDE: Configuring the AutoFS **-hosts** Map 10-13

© 2003 Hewlett-Packard Development Company, L.P.

10–7. SLIDE: Configuring the AutoFS Direct Map 10-16
10–8. SLIDE: Configuring the AutoFS Indirect Maps....10-19
10–9. SLIDE: Comparing Direct versus Indirect Maps 10-22
10–10. SLIDE: Mounting Home Directories with AutoFS....10-24
10–11. SLIDE: Mounting Home Directories with AutoFS Key Substitution....10-27
10–12. SLIDE: Configuring AutoFS to Access Replicated Servers....10-29
10–13. SLIDE: Troubleshooting AutoFS....10-31
10–14. SLIDE: Comparing AutoFS with Automounter....10-34
10–15. LAB: Configuring AutoFS....10-36

Module 11 — Configuring NIS
11–1. SLIDE: Why Use NIS? 11-2
11–2. SLIDE: NIS Maps 11-4
11–3. SLIDE: NIS Domains....11-6
11–4. SLIDE: NIS Roles....11-7
11–5. SLIDE: NIS Startup Files 11-8
11–6. SLIDE: NIS Daemons 11-10
11–7. SLIDE: Configuring NIS Servers and Clients 11-12
11–8. SLIDE: Testing NIS....11-14
11–9. SLIDE: Changing Passwords on an NIS Node 11-16
11–10. SLIDE: Updating and Propagating Maps on the Master Server....11-18
11–11. SLIDE: Fetching Maps from the Master Server 11-20
11–12. SLIDE: Restricting Access to NIS Clients and Slave Servers....11-23
11–13. SLIDE: Restricting Access to the Master Server 11-26
11–14. LAB: Configuring NIS....11-28

Module 12 — Configuring DNS
12–1. SLIDE: Resolving Host Names to IP Addresses 12-2
12–2. SLIDE: DNS Overview 12-4
12–3. SLIDE: The DNS Hierarchical Name Space 12-6
12–4. SLIDE: Public and Private Name Spaces....12-8
12–5. SLIDE: **`in-addr.arpa`** Name Space....12-10
12–6. SLIDE: DNS Name Servers....12-12
12–7. SLIDE: DNS Name Server Zones....12-13
12–8. SLIDE: Resolving Host Names in the Local Domain....12-15
12–9. SLIDE: Resolving Host Names in Other Domains....12-17
12–10. SLIDE: Configuring a Master Server....12-19
12–11. SLIDE: Configuring a Slave Server....12-21
12–12. SLIDE: Configuring a Cache-Only Name Server....12-24
12–13. SLIDE: Testing Name Servers with **`nslookup`**....12-26
12–14. SLIDE: Configuring DNS Clients....12-28
12–15. SLIDE: Configuring the Name Service Switch....12-31
12–16. SLIDE: Testing Resolvers with **`nsquery`** 12-36
12–17. SLIDE: Introducing **`/etc/named.data`**....12-38
12–18. SLIDE: Introducing **`/etc/named.conf`**....12-40
12–19. SLIDE: Loading the DNS Data Files....12-42
12–20. SLIDE: Updating the Primary Server 12-43
12–21. SLIDE: Updating the Secondary Server....12-45
12–22. LAB: Configuring DNS....12-47

© 2003 Hewlett-Packard Development Company, L.P.

Module 13 — Configuring the ARPA/Berkeley Services

13–1. SLIDE: Internet Services Overview 13-2
13–2. SLIDE: Internet Service Clients and Servers 13-5
13–3. SLIDE: Starting Internet Services via **/sbin/rc** 13-7
13–4. SLIDE: Starting Internet Services via **inetd** 13-8
13–5. SLIDE: Configuring **/etc/inetd.conf** 13-10
13–6. SLIDE: Configuring **/etc/services** 13-12
13–7. SLIDE: Configuring **/var/adm/inetd.sec** 13-15
13–8. SLIDE: Configuring **inetd** Logging 13-17
13–9. SLIDE: System and User Equivalency 13-19
13–10. SLIDE: Configuring **/etc/hosts.equiv** 13-20
13–11. SLIDE: Configuring **~/.rhosts** 13-22
13–12. SLIDE: FTP Configuration Issues 13-24
13–13. SLIDE: ARPA/Berkeley Services Review 13-27
13–14. LAB: Configuring and Securing ARPA/Berkeley Services 13-29
13–15. REVIEW QUESTIONS: Configuring and Securing ARPA/Berkeley Services 13-40

Module 14 — Configuring a BOOTP/TFTP Server

14–1. SLIDE: What Are **bootp** and **tftp?** 14-2
14–2. SLIDE: Enabling **bootp** and **tftp** Services 14-3
14–3. SLIDE: Configuring **/etc/bootptab** 14-5
14–4. SLIDE: Booting Network Printers 14-7
14–5. SLIDE: Configuring a Network Printer Server 14-8
14–6. SLIDE: What Is an X-Station? 14-10
14–7. SLIDE: Booting X-Station 14-12
14–8. SLIDE: Configuring an X-Terminal Server 14-16
14–9. LAB: Managing a **bootp/tftp** Server 14-21

Module 15 — Configuring NTP

15–1. SLIDE: Introduction to the Network Time Protocol (NTP) 15-2
15–2. SLIDE: NTP Time Sources 15-4
15–3. SLIDE: NTP Stratum Levels 15-5
15–4. SLIDE: NTP Roles 15-7
15–5. SLIDE: Defining NTP Servers via **/etc/ntp.conf** 15-9
15–6. SLIDE: Defining NTP Clients via **/etc/ntp.conf** 15-11
15–7. SLIDE: How NTP Adjusts the System Clock 15-13
15–8. SLIDE: Configuring an NTP Server 15-15
15–9. SLIDE: Configuring an NTP Client 15-17
15–10. SLIDE: Verifying NTP Functionality 15-19
15–11. LAB: Introduction to NTP 15-21

Module 16 — Configuring an SD-UX Server

16–1. SLIDE: Why Create an SD-UX Depot Server? 16-2
16–2. SLIDE: SD-UX Concepts 16-4
16–3. SLIDE: Managing Depots 16-5
16–4. SLIDE: Listing Depots and Products 16-7
16–5. SLIDE: Installing Products from a Depot 16-10
16–6. SLIDE: Auditing Depot Usage 16-11
16–7. LAB: Creating and Managing an SD-UX Depot 16-13

© 2003 Hewlett-Packard Development Company, L.P.

Appendix A — Decimal-Hexadecimal-Binary Conversion

Appendix B — HP-UX Administration Command Quick Reference

Solutions

© 2003 Hewlett-Packard Development Company, L.P.

Overview

Course Description

This course is targeted at the HP-UX system administrator who must configure and administer HP-UX 10.X or 11.00 systems in an IEEE 802.3 local area network and be responsible for HP-UX network administration. This course was updated to include HP-UX 11.0 material, but still applies to 10.x systems. Differences between the two operating systems are specified in the student notes sections.

Student Performance Objectives

Module 1 — LAN Concepts

- Describe the purpose of a local area network (LAN).
- Describe the concept and purpose of the OSI model.
- Describe the role of host names, IPs, MACs, ports, and sockets in the OSI model.
- Describe the format and purpose of a MAC address.
- Describe the format and purpose of an IP address.
- Describe the format and purpose of an IP netmask.
- Describe the format and purpose of an IP network address.
- Describe the format and purpose of an IP broadcast address.
- Describe the format and purpose of the IP loopback address.
- Describe the format and purpose of a host name.
- Describe the differences between the UDP and TCP protocols.
- Describe the purpose of ports and sockets.
- Describe the host name to IP to MAC address lookup process.

Module 2 — LAN Hardware Overview

- Describe the characteristics of three major LAN cable types.
- Discuss three different LAN topologies.
- Explain two different LAN access methods.
- List the characteristics of an Ethernet LAN.

© 2003 Hewlett-Packard Development Company, L.P.

- List the characteristics of a Token Ring LAN.
- List the characteristics of an FDDI LAN.
- Explain the difference between physical and logical topologies.
- Describe the role of repeaters, hubs, bridges, switches, routers, gateways, and firewalls in a local area network.

Module 3 — Configuring IP Connectivity

- Configure software and drivers to support a newly installed network interface card.
- Configure link layer connectivity with the **`lanadmin`** command.
- Configure and view the system host name with the **`hostname`** command.
- Configure and view the system IP address and netmask with the **`ifconfig`** command.
- Configure IP multiplexing.
- Configure and use the **`/etc/rc.config.d/netconf`** configuration file.
- Configure the **`/etc/hosts`** configuration file.

Module 4 — Configuring IP Routing

- Configure static routes.
- Configure a default route.
- View the routing tables.

Module 5 — Configuring Subnetting

- List the advantages and disadvantages of a subnetted network.
- Subnet a network on an octet boundary.
- Subnet a network on a non-octet boundary.
- Set an HP-UX subnet mask.

© 2003 Hewlett-Packard Development Company, L.P.

Module 6 — Troubleshooting Network Connectivity

- Use the following tools to troubleshoot network connectivity:
  - **lanscan**
  - **lanadmin**
  - **linkloop**
  - **arp/ndd**
  - **ping**
  - **netstat -i**
  - **netstat -a**
  - **netstat -r**
  - **hostname**
  - **nslookup**

Module 7 — Starting Network Services

- Describe how run levels are used during system boot time.
- Change and view the system's current run level.
- Define the default system run level.
- Enable/disable services via the **/etc/rc.config.d config** files.
- Create custom startup and shutdown scripts to start additional services during the boot process.
- View the startup error log file.

Module 8 — NFS Concepts

- Describe the purpose and function of NFS.
- Define *NFS server* and *NFS client*.
- List probable candidates for file sharing via NFS.
- Describe the purpose of NFS RPCs.
- Describe the purpose of the **portmap** and **rpcbind** daemons.
- Compare and contrast the NFS PV2 and NFS PV3 protocols.
- Compare and contrast the NFS and CIFS protocols.

© 2003 Hewlett-Packard Development Company, L.P.

Module 9 — Configuring NFS

- Configure NFS server functionality.
- Export file systems, and determine access privileges for those file systems.
- Configure NFS client functionality.
- Mount and unmount NFS file systems.
- Automatically mount NFS file systems.
- Determine which file systems have been exported and mounted.
- Describe the NFS startup procedure.
- Describe the function of the following NFS configuration files:

  `/etc/rc.config.d/nfsconf`
  `/etc/exports`
- List the daemons that must be running on an NFS server and client.
- Use **`showmount`**, **`rpcinfo`**, and **`nfsstat`** to troubleshoot problems with NFS.

Module 10 — Configuring AutoFS

- Describe the reasons for using AutoFS.
- Start and stop the AutoFS daemons.
- Configure the AutoFS master map.
- Configure the AutoFS **`-hosts`** special map.
- Configure the AutoFS direct map.
- Configure the AutoFS indirect maps.
- Describe the differences between AutoFS direct and indirect maps.
- Configure AutoFS to mount and unmount user home directories.
- Troubleshoot problems with AutoFS.
- Identify the limitations of AutoFS's predecessor, the NFS Automounter.

© 2003 Hewlett-Packard Development Company, L.P.

Module 11 — Configuring NIS

- Describe the purpose of Network Information Service (NIS).
- List the standard NIS maps.
- Configure an NIS master server.
- Configure an NIS slave server.
- Configure an NIS client.
- Change a password stored in the password map.
- Update other NIS maps on the master server.
- Propagate new maps to a slave server.
- Restrict user access to the master server.
- Describe the differences between NIS and NIS+.

Module 12 — Configuring DNS

- Compare and contrast the three approaches to host name resolution:
  - **`/etc/hosts`**
  - NIS
  - DNS/BIND
- Configure a primary DNS server using the **`hosts_to_named`** command.
- Configure a secondary name server.
- Configure a cache-only name server.
- Configure a resolver-only host.
- Configure the **`/etc/nsswitch.conf`** file.
- Add or remove a host in the DNS database, using the **`hosts_to_named`** command.
- Troubleshoot DNS using **`nslookup`** and **`nsquery`**.
- Describe the purpose and format of the following configuration files:
  - **`/etc/rc.config.d/namesvrs`**
  - **`/etc/named.conf`**

© 2003 Hewlett-Packard Development Company, L.P.

- **/etc/resolv.conf**

Module 13 — Configuring the ARPA/Berkeley Services

- List the commonly used ARPA-Berkeley services.
- Describe the function of the Internet daemon, **inetd**.
- Describe the process used to request **ftp/telnet** service from **inetd**.
- Describe the Internet service configuration files.
- Enable or disable Internet services from the command line.
- Allow or prevent access to selected Internet services via the **inetd.conf** file.
- Allow or prevent access for selected clients via the **inetd.sec** file.
- Allow or prevent access for selected users via the **passwd** file.
- Log requests for ARPA/Berkeley services.
- Define host equivalency between hosts with the **/etc/hosts.equiv** file.
- Define user equivalency between hosts with the **~/.rhosts** file.

Module 14 — Configuring a BOOTP/TFTP Server

- Describe the purpose of **bootp** and **tftp**.
- Configure **inetd** to provide **bootp** and **tftp** services.
- Describe the purpose and contents of the **bootptab** file.
- Describe the purpose of a network-based printer.
- Configure a **bootptab** entry for a network printer using **hppi**.
- Configure a **bootptab** entry for an X terminal using **xtadm**

Module 15 — Configuring NTP

- List three reasons for implementing network time synchronization.
- Describe the NTP **stratum level** concept.

© 2003 Hewlett-Packard Development Company, L.P.

- Define the following terms:
  - NTP server
  - NTP peer
  - NTP broadcast client
  - NTP polling client
- Configure an NTP server.
- Configure an NTP broadcast client.
- Configure an NTP direct-poll client.
- Monitor NTP using the **`ntpq`** command

Module 16 — Configuring an SD-UX Server

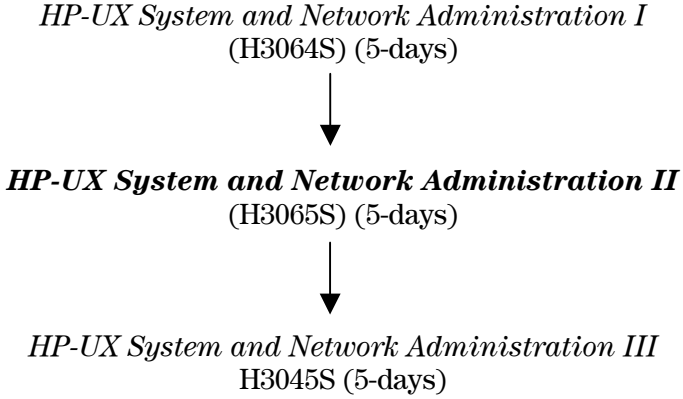
- Create an SD-UX server.
- Copy software to an SD-UX server.
- Register an SD-UX server.
- Audit depot usage.

Student Profile and Prerequisites

This course is designed for the student who is responsible for administering both systems and networks in an HP-UX environment. HP 9000 Series 300/400, workstation, and server tasks are covered. The student should be an experienced HP-UX system administrator. The student should have completed the following course:

- *HP-UX System and Network Administration I* (H3064S)

Curriculum Path



© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 1 — LAN Concepts

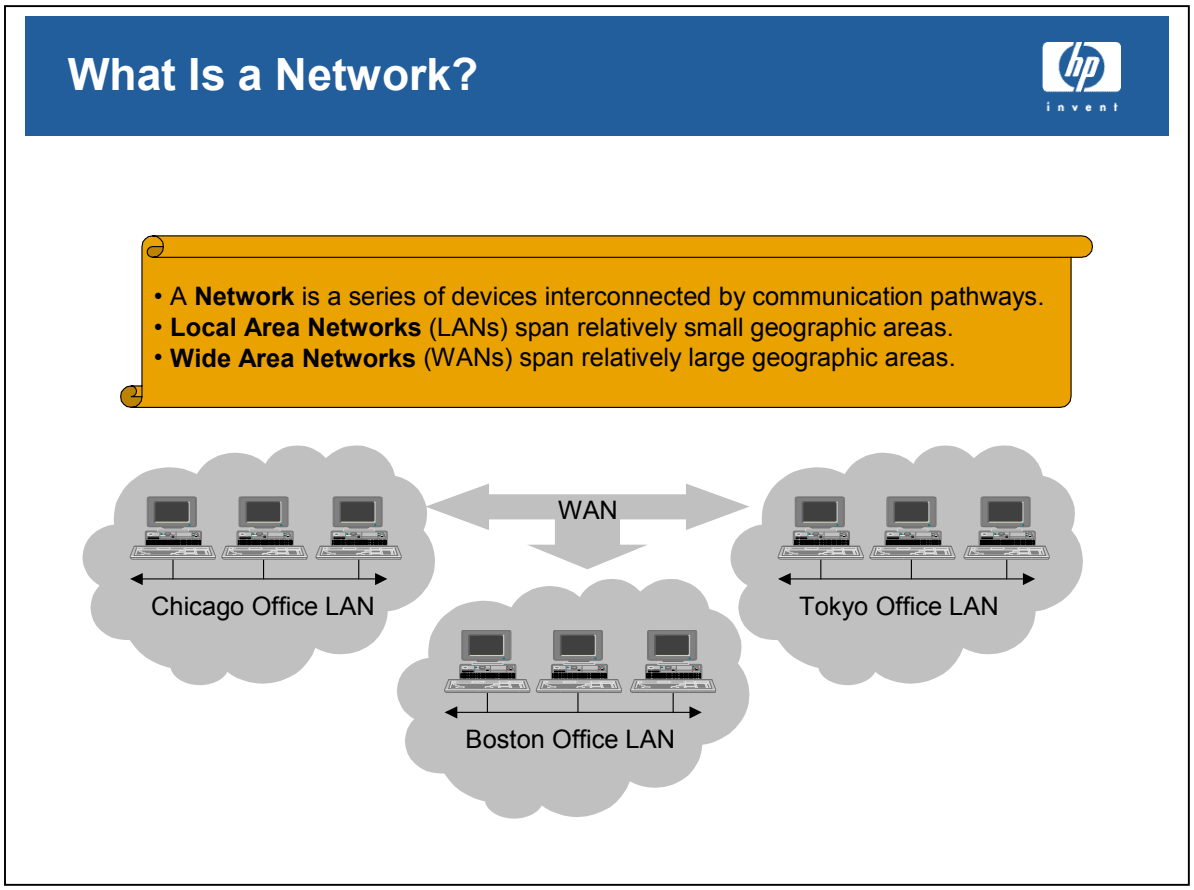
Objectives

Upon completion of this module, you will be able to do the following:

- Describe the purpose of a local area network (LAN).
- Describe the concept and purpose of the OSI model.
- Describe the role of host names, IPs, MACs, ports, and sockets in the OSI model.
- Describe the format and purpose of a MAC address.
- Describe the format and purpose of an IP address.
- Describe the format and purpose of an IP netmask.
- Describe the format and purpose of an IP network address.
- Describe the format and purpose of an IP broadcast address.
- Describe the format and purpose of the IP loopback address.
- Describe the format and purpose of a host name.
- Describe the differences between the UDP and TCP protocols.
- Describe the purpose of ports and sockets.
- Describe the host name to IP to MAC address lookup process.

© 2003 Hewlett-Packard Development Company, L.P.

1–1. SLIDE: What Is a Network?



Student Notes

The System and Network Administration I course that preceded this class dealt primarily with administration issues on a single system. This course will concentrate on the technologies and services used to share resources among multiple UNIX hosts on a computer network. Perhaps we should start with some definitions.

What Is a Computer Network?

A **Computer Network** is simply a collection of systems and devices interconnected by some sort of data pathway for the purpose of sharing resources. Many different types of resources may be shared across a computer network. For instance:

- Few systems these days have a dedicated, locally attached printer. Oftentimes, multiple systems share one or more network **printers**.
- Disk resources may be shared via a network, too. Many users access files, directories, and even executables via network **file servers**.
- If your desktop computer does not have a tape drive, you may choose to write system backups to a tape drive physically attached to a **tape backup server** host elsewhere on your network.
- Even CPU resources may be shared via a network. Users may run a simple executable on a desktop system that queries a **database server** across the network.

© 2003 Hewlett-Packard Development Company, L.P.

Local Area Networks versus Wide Area Networks

Networks are often categorized as Local Area Networks (LANs) or Wide Area Networks (WANs).

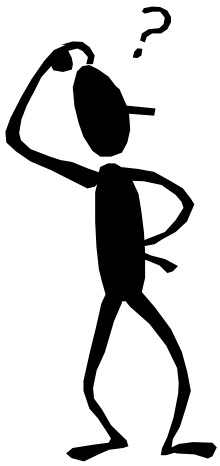
HP officially defines a **local area network (LAN)** as a network that transmits a large amount of information at a relatively high speed over limited distances within a single facility or site. For instance, devices within a branch office are oftentimes connected via a local area network. In a larger organization, each department may have a separate, dedicated LAN.

A **wide area network (WAN)** is a network that covers a large geographic area, allowing devices in different cities to communicate with one another, though often at a data transmission rate that is much slower than a LAN. Oftentimes, multiple LANs are connected together via a WAN. Types of well-known WANs include the ARPANET and the public X.25 network.

© 2003 Hewlett-Packard Development Company, L.P.

1–2. SLIDE: The OSI Model in a Nutshell

The OSI Model in a Nutshell



7	Application	How is data created and used?
6	Presentation	How is the data represented to the application? Is the data in EBCDIC or ASCII format?
5	Session	How does an application initiate a connection? How does an application actually transmit/receive data? How does an application know data has been received?
4	Transport	Should the receiver acknowledge receipt of a packet? How should the acknowledgement be handled? Which process should receive the data?
3	Network	How is data routed between networks?
2	Data link	How do I know when its my turn to transmit? How do I know which data is for me? How are collisions handled?
1	Physical	What kinds of cabling are supported? What kinds of connectors are supported? What's the longest supported cable segment?

Student Notes

Because no single vendor can meet the needs of the entire networking marketplace, companies have to draw on multiple vendors for their communications hardware and software. The unique network architectures and proprietary protocols developed by each vendor are frequently incompatible, precluding communication among them. The Open Systems Interconnection (OSI) model was developed by the International Standards Organization to resolve these incompatibility issues and allow products from different manufacturers to communicate with one another.

The layer concept, on which the OSI model is based, establishes a set of rules for data transmission on a variety of levels. In the layered scheme, messages originate from the top layer (layer 7) of a transmitting computer, move down to its lowest layer (layer 1), and travel across the network media to the receiving computer. The message arrives at the lowest layer of the receiving computer (layer 1), and moves up through its various layers to layer 7.

The following describes each layer in detail:

- **Layer 7: The application layer** provides the software for network services such as file transfer, remote login, remote execution, and electronic mail. It provides the interface between user programs and the network.
  "What the user runs"

© 2003 Hewlett-Packard Development Company, L.P.

- **Layer 6: The presentation layer** converts outbound data from a machine-specific format to an international standard format. It converts inbound back to a machine-specific format (for example: ASCII -> machine specific -> EBCDIC).
  "Translator"

- **Layer 5: The session layer** allows the setup and termination of a communications path and synchronizes the dialog between the two systems. It establishes connections between systems in much the same way as an automatic dialer does between two telephone systems.
  "Terminal emulator"

- **Layer 4: The transport layer** provides reliable flow of datagrams between sender and receiver, and ensures that the data arrives at the correct destination. Protocols at this layer also ensure that a copy of the data is made in case it is lost in transmission.
  "Software error correction"

- **Layer 3: The network layer** decides which path will be taken through the network. It provides the packet addressing that will tell computers on the network where to route the user's data.
  "Addressing scheme"

- **Layer 2: The data link layer** provides reliable, error-free media access for data transmission. It produces the frame around the data.
  "Hardware error correction"

- **Layer 1: the physical layer** establishes the actual physical connection (cable connection) between the network and the computer equipment. Physical Layer standards determine what type of signaling is used (what represents a bit 0, what represents a 1), what cable types and lengths are supported, and what types of connectors may be used.
  "Cable"

© 2003 Hewlett-Packard Development Company, L.P.

1–3. TEXT PAGE: OSI Worksheet

Table 1

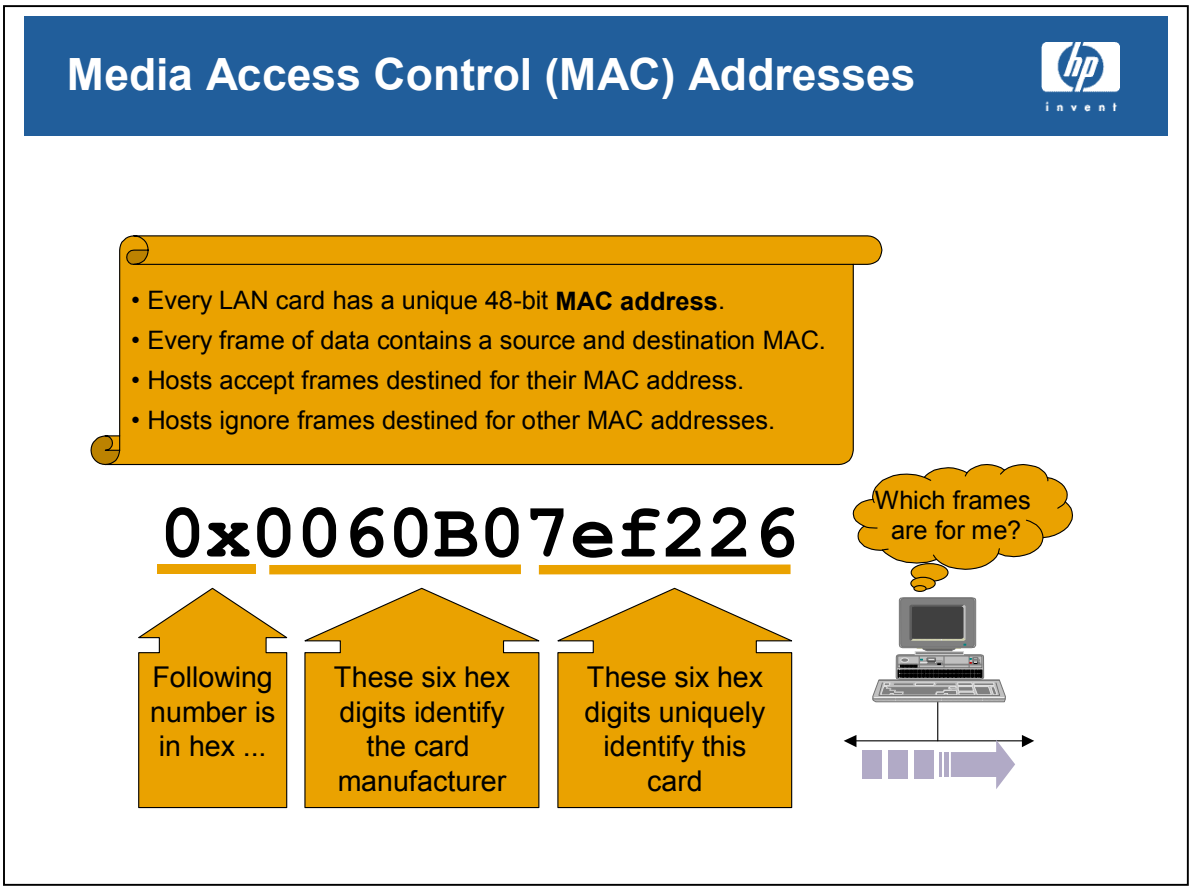
OSI Layer	Associated Protocols and Addresses
7	
6	
5	
4	
3	
2	
1	

Instructions

The remainder of this chapter provides an overview of the protocols and network address types that are required to pass data across a network from one process to another. As new protocols and network address types are introduced, record them in the appropriate layer of this OSI chart.

© 2003 Hewlett-Packard Development Company, L.P.

1–4. SLIDE: Media Access Control (MAC) Addresses



Student Notes

In order to pass data successfully from host to host on a local area network, there must be some mechanism for determining which frames of data are destined for which hosts. **Media Access Control** addresses solve this problem!

Every LAN card attached to a local area network must have a unique MAC address assigned to it. Every frame of data passed across the network, then, includes both a source and destination MAC address. If the destination MAC address on a passing frame matches a host's own MAC address, the host knows that it should receive that frame of data. Frames destined for other MAC addresses are ignored. While you may be accustomed to referencing hosts on the network by "host name" or "IP address," those addresses must be mapped to MAC addresses before a frame of data can be sent across the network wire. Host names and IP addresses will be discussed in detail later in this chapter.

The MAC address is a 48-bit number that is set by the LAN card manufacturer. Typically, HP-UX displays the MAC address as a 12-digit hexadecimal number, preceded by a 0x to indicate that the value is in hex. The first six hexadecimal digits indicate which manufacturer produced the card, while the last six digits uniquely distinguish each card produced by that manufacturer from all others. Currently, HP LAN card MAC addresses begin with 0x080009 or 0x0060b0. The MAC address may be changed via the **`lanadmin`** command, but this is not recommended.

© 2003 Hewlett-Packard Development Company, L.P.

Viewing a Host's MAC Addresses

If you have multiple LAN cards, each LAN card should have a different MAC address. Use the **lanscan** command to view your system's MAC addresses. The following example shows **lanscan** output for a host with two network interface cards:

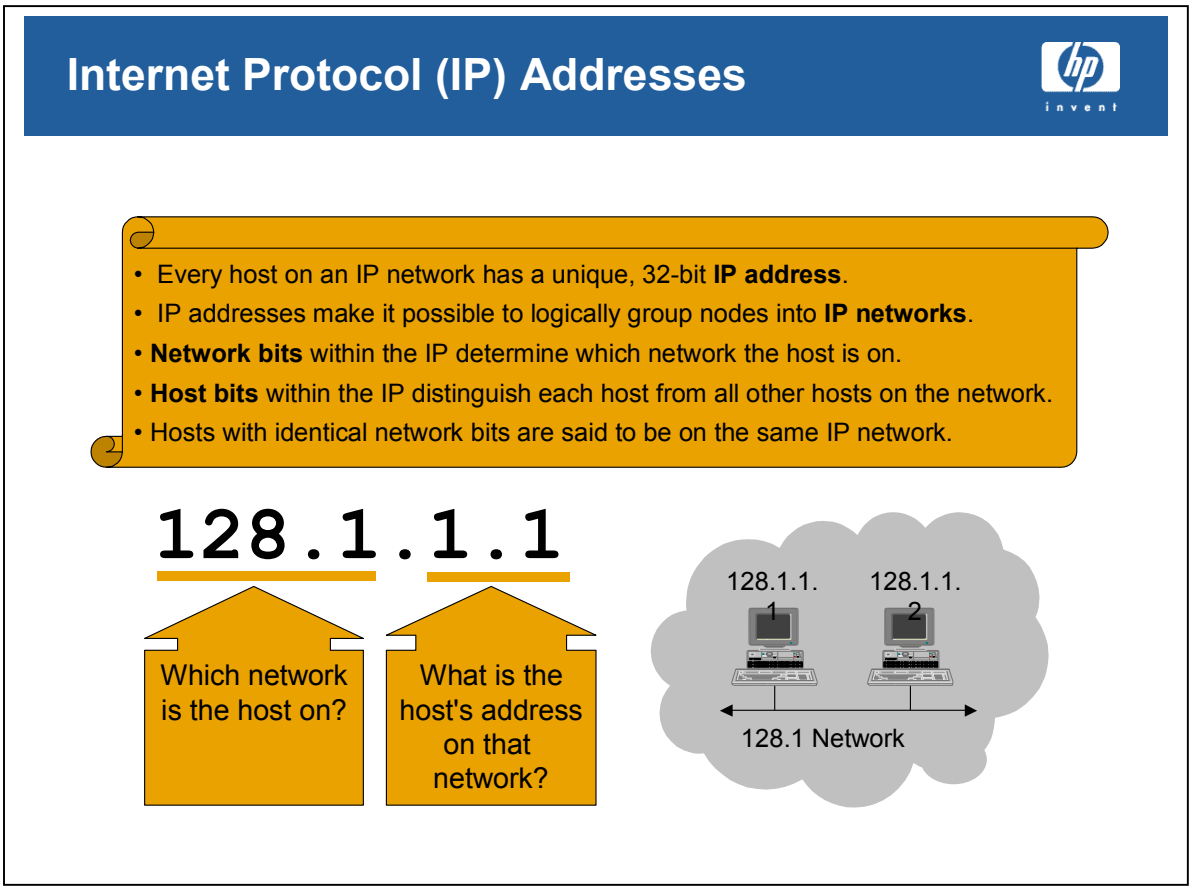
```
# lanscan
Hardware Station        Crd Hdw   Net-Interface  NM  MAC   HP-DLPI DLPI
Path     Address        In# State NamePPA        ID  Type  Support Mjr#
2/0/2    0x0800094A7334 0   UP    lan0 snap0     1   ETHER Yes     119
4/0/1    0x080009707AF2 1   UP    lan1 snap1     2   ETHER Yes     119
```

NOTE: The MAC address is often referenced via a variety of different names. All of these names refer to the same address:

- *link-level* address
- *station* address
- *physical* address
- *hardware* address
- *Ethernet* address

© 2003 Hewlett-Packard Development Company, L.P.

1–5. SLIDE: Internet Protocol (IP) Addresses



Student Notes

In addition to the MAC address assigned to each LAN card by the card manufacturer, each LAN card on an HP-UX machine is also typically assigned an Internet Protocol (IP) Address.

Internet Protocol Addresses (or **IP Addresses**) make it possible to group nodes into logical IP networks, and efficiently pass data between these networks. For instance, hosts within your Chicago office may all be assigned IP addresses on one IP network, while hosts in your San Francisco office may be assigned IP addresses on a different IP network. By looking at a data packet's destination IP address, your network devices can intelligently "route" data between networks.

IP Address Structure

IP addresses are usually represented by four 8-bit fields, separated by dots ("."). These fields are called *octets*. Each 8-bit octet is represented by a decimal number in the range from 0 to 255.

© 2003 Hewlett-Packard Development Company, L.P.

The table below demonstrates the conversion of several 8-bit binary numbers to their corresponding decimal values:

128	64	32	16	8	4	2	1	Decimal Value
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	2
0	0	0	0	0	0	1	1	3
0	0	0	0	0	1	0	0	4
0	0	0	0	0	1	0	1	5
1	1	1	1	1	1	1	1	255

Using this conversion mechanism, IP addresses may be displayed in either binary or decimal. Consider the following examples:

```
10000000.00000001.00000001.00000001 = 128.1.1.1
10001010.10000001.00000001.00000010 = 138.129.1.2
10011100.10011011.11000010.10101010 = 156.153.194.170
```

IP Address Network and Host Bits

Some bits within an IP address identify the network to which the host belongs. These **network bits** are used by network devices to route data between networks. Two hosts with identical network bits are said to be on the same IP network.

The remaining **host bits** in the IP address uniquely identify each host within the logical network.

Viewing a Host's IP Address

You can view your system's IP addresses with two commands. First, use the **`lanscan`** command that was introduced on the previous slide to determine the "Interface Name" that has been assigned to each of your LAN cards:

```
# lanscan
Hardware Station        Crd Hdw   Net-Interface  NM  MAC   HP-DLPI DLPI
Path     Address        In# State NamePPA        ID  Type  Support Mjr#
2/0/2    0x0800094A7334 0   UP    lan0 snap0     1   ETHER Yes     119
4/0/1    0x080009707AF2 1   UP    lan1 snap1     2   ETHER Yes     119
```

Next, use the **`ifconfig`** command to view each LAN card's IP address:

```
# ifconfig lan0
lan0: flags=843<Up,BROADCAST,RUNNING,MULTICAST>
      inet 128.1.1.1 netmask ffff0000 broadcast 128.1.255.255
```


© 2003 Hewlett-Packard Development Company, L.P.

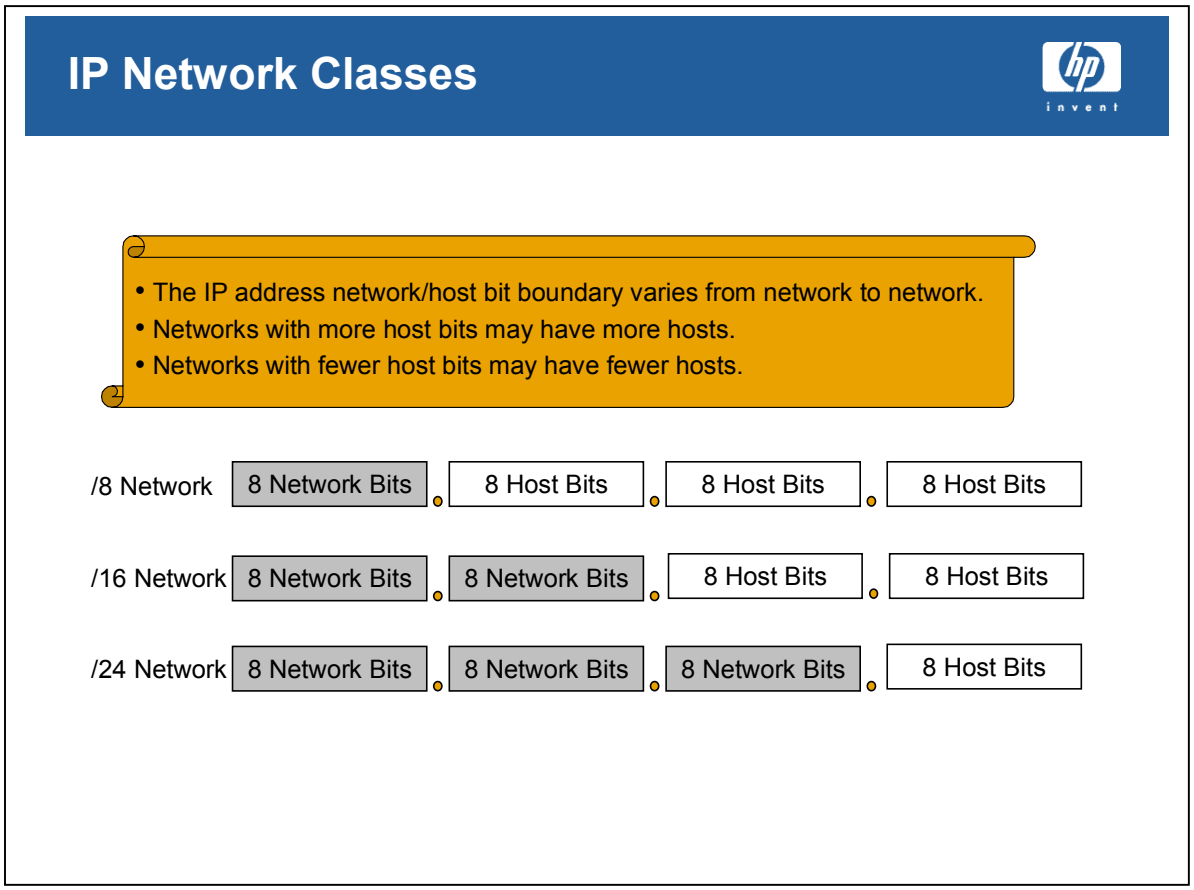
The **netstat** command can also be used to display your IP address:

```
# netstat -in
Name    Mtu  Network     Address     Ipkts   Opkts
lan0    1500 128.1.0.0   128.1.1.1   55670   23469
lo0     4136 127.0.0.0   127.0.0.1    3068    3068
```

CAUTION: Do not assign the same IP address to different hosts. If two hosts on the same network use the same IP address, errors will occur when communicating with these hosts.

© 2003 Hewlett-Packard Development Company, L.P.

1–6. SLIDE: IP Network Classes



Student Notes

The previous slide noted that IP addresses have two components: a network component and a host component. The original designers of the Internet realized that some networks would be very large, while others would be much smaller. Large networks would require more host bits to provide a unique host address for each node, while smaller networks would require fewer host bits to provide a unique host address for each node.

Varying the IP address network/host boundary makes it possible to allocate just enough IP addresses for any size network. Thus, although every IP address is 32 bits, the boundary between the network and host portions of an IP address varies from network to network.

When your ISP or IT department assigns you an IP address, the IP will often have a ***/xx*** appended to the end. The ***/xx*** identifies the number of network bits in the IP address.

© 2003 Hewlett-Packard Development Company, L.P.

The following table demonstrates the effect of shifting the network boundary. The table only shows /8, /16, and /24 networks; many others are possible, too.

Network Type	Network bits	Host bits	Host Addresses/ Network
/8	8	24	2^{24} = 16,777,216
/16	16	16	2^{16} = 65,536
/24	24	8	2^{8} = 256

** Note: Not all of the host addresses are actually usable. One of the addresses in each network is used as the network address, another is used as the broadcast address. Thus, there can only be 254 hosts on a /24 network. These special addresses will be discussed later.

Traditional Class A, B, and C IP Addressing

In the early days of the Internet, only three types of networks were recognized: /8 (also known as "Class A") networks, /16 (also known as "Class B") networks, and /24 (also known as "Class C") networks. Large organizations were assigned "Class A" network addresses, medium sized organizations were assigned "Class B" network addresses, and smaller organizations were assigned "Class C" network addresses.

Furthermore, the addresses were structured such that network devices could determine an IP address's class (and network/host boundary!) by simply looking at the first few bits:

- Any IP address beginning with a binary "0" was assumed to be a Class A.
  In decimal notation, these IP addresses have a number between 1 and 127 in octet 1.
- Any IP address beginning with a binary "10" was assumed to be a Class B.
  In decimal notation, these IP addresses have a number between 128 and 191 in octet 1.
- Any IP address beginning with a binary "110" was assumed to be a Class C.
  In decimal notation, these IP addresses have a number between 192 and 223 in octet 1.

The following chart summarizes the resulting network classes.

Class	Net bits	Host bits	Number of Networks	Hosts / Network	Range
Class A	8	24	127	16,777,216	1–127
Class B	16	16	16,383	65,536	128–191
Class C	24	8	2,097,151	256	192–223

Unfortunately, the Class A/B/C IP allocation scheme led to inefficient use of the IP address space, since many organizations were given much larger IP address blocks than they actually needed. HP, for instance, was assigned Class A address 15.0.0.0/8. This address space includes over 16 million IP addresses! This largesse was not considered a problem at the time, since there seemed to be far more addresses than would ever be used. No one anticipated the tremendous growth in the Internet that has occurred over the last decade.

© 2003 Hewlett-Packard Development Company, L.P.

In the 1990s, the Internet Engineering Task Force (IETF) committee decided to move to the more flexible scheme known as Classless Internet Domain Routing (CIDR) that is used today. Now you may be assigned a /13, /14, /15, /16, /23 — or almost any other network type — depending on the number of hosts on your network.

Furthermore, using the new "Classless" IP addressing scheme, you may find that your IP address is 192.1.1.1/20. Using the older "Classfull" IP addressing scheme, any IP beginning with 192 had to be a Class C with 24 network bits. The new scheme is more flexible, but also somewhat more complicated.

IPv6 Addressing

CIDR addressing and other creative solutions have made it possible to more efficiently use the existing 32-bit IP address space more efficiently. However, a 32-bit address can represent at most 2^{32} (about 4 billion) addresses, and as more and more devices attach to the Internet, this address space is being rapidly depleted.

As far back as 1991, the Internet Engineering Task Force began considering a successor to the current 32-bit, 4-octet "IPv4" addressing method. After nearly a decade of study and debate, the IETF has settled on a new standard which has been dubbed "IPv6". The new IPv6 standard uses a 128-bit addressing scheme to exponentially increase the pool of IP addresses. Unfortunately, IPv6 addresses are also much more cumbersome than our current IPv4 addresses; they are typically represented as a series of eight four digit hexadecimal numbers. Here's a typical IPv6 address:

```
CDCD:910A:2222:5498:8475:1111:3900:2020
```

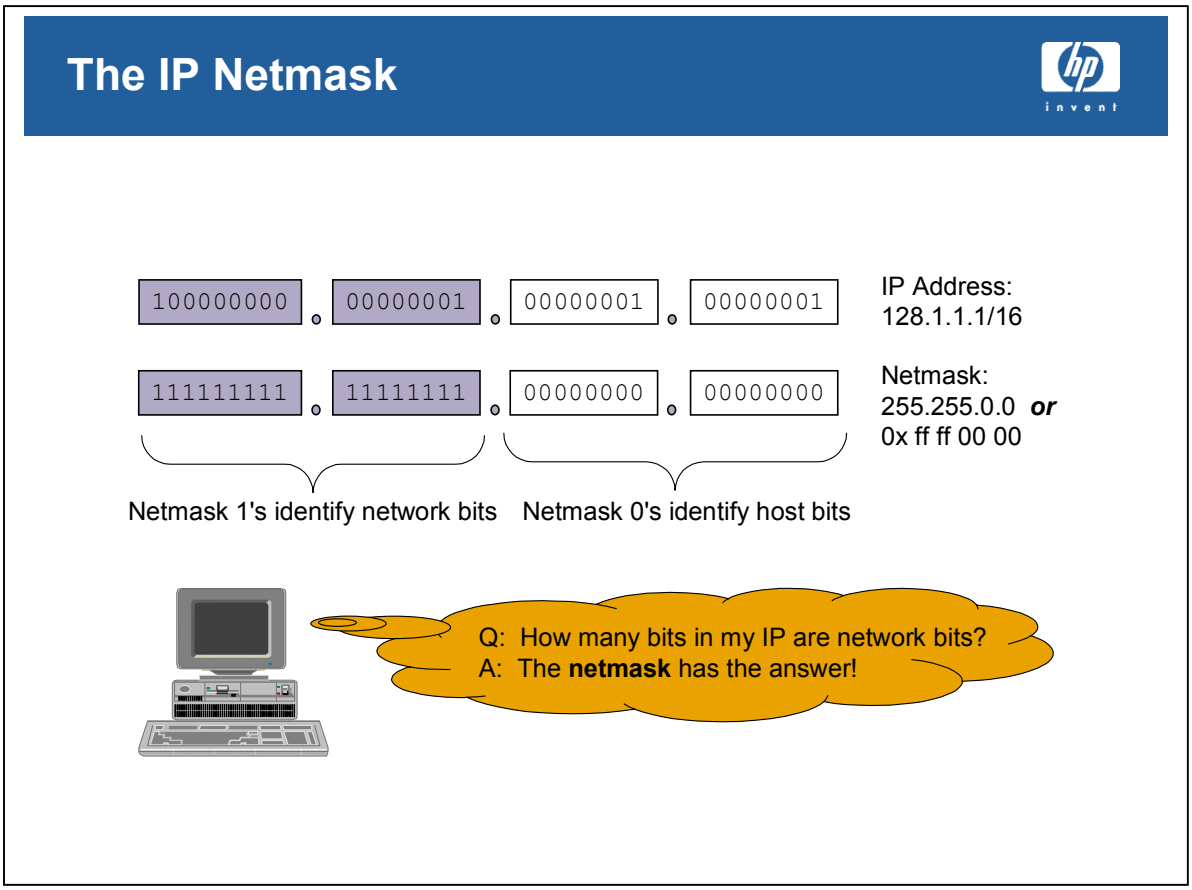
Fortunately, the transition to IPv6 needn't occur overnight. As long as all the hosts on your local area network continue to use IPv4, there is no need to upgrade your servers and workstations to IPv6. The overall transition from IPv4 to IPv6 is expected to proceed gradually over the course of several years.

HP currently offers an IPv6 developers' toolkit, but full support for IPv6 on HP-UX won't be available until a future release of the OS.

For more information on IPv6, take a look at Pete Loshin's *IPv6 Clearly Explained* (ISBN 0124558380), or Christian Huitema's more technical *IPv6: the New Internet Protocol* (ISBN 0138505055).

© 2003 Hewlett-Packard Development Company, L.P.

1–7. SLIDE: The IP Netmask



Student Notes

When you configure your system's IP address, your system must be told which bits in your IP address are network bits, and which bits are host bits. These days, the network/host boundary is usually communicated via the "/" notation introduced on the previous page. However, UNIX uses a different mechanism to identify the network/host boundary: the IP netmask.

The netmask, like an IP address, has 32 bits. However, the netmask is formulated somewhat differently than a standard IP address. To determine your netmask, write a "1" in each network bit, and a "0" in each of the remaining bits. The resulting value may be written in binary, dotted-decimal (like an IP address), or even in hexadecimal. The chart below shows some common netmasks in all three forms:

Net Type	Netmask		
	Netmask (Binary)	(Hex)	(Decimal)
/8	11111111.00000000.00000000.00000000	0xff000000	255.0.0.0
/16	11111111.11111111.00000000.00000000	0xffff0000	255.255.0.0
/24	11111111.11111111.11111111.00000000	0xffffff00	255.255.255.0

For other conversions, either consult the binary/hex/decimal conversion chart at the end of this book, or use the **`/usr/dt/bin/dtcalc`** calculator utility.

© 2003 Hewlett-Packard Development Company, L.P.

Viewing Your System's IP Netmask

You can view your system's IP netmask with the **ifconfig** command. First, use the **lanscan** command to determine the "Interface Name" that has been assigned to each of your LAN cards:

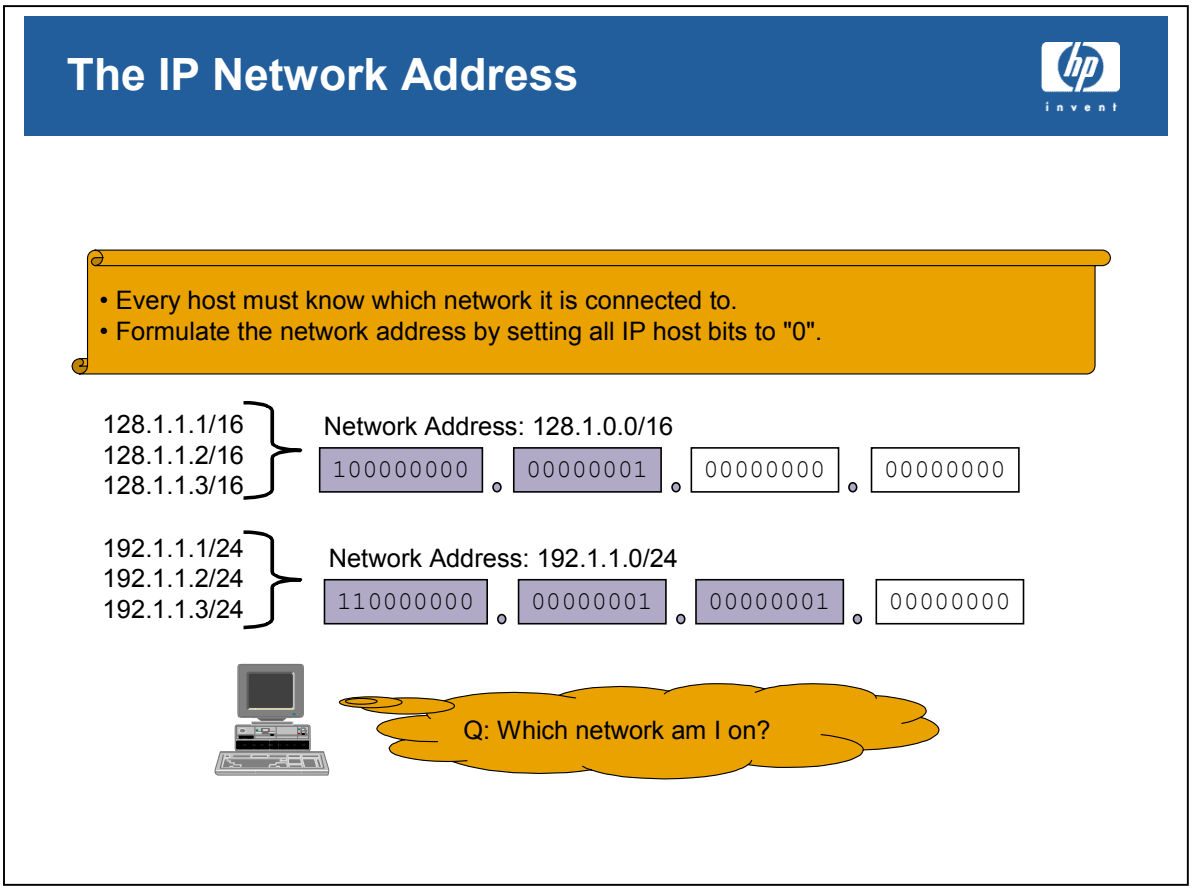
```
# lanscan
Hardware Station        Crd Hdw   Net-Interface  NM  MAC   HP-DLPI DLPI
Path     Address        In# State NamePPA        ID  Type  Support Mjr#
2/0/2    0x0800094A7334 0   UP    lan0 snap0     1   ETHER Yes     119
4/0/1    0x080009707AF2 1   UP    lan1 snap1     2   ETHER Yes     119
```

Next, use the **ifconfig** command to view each LAN card's netmask:

```
# ifconfig lan0
lan0: flags=843<Up,BROADCAST,RUNNING,MULTICAST>
      inet 128.1.1.1 netmask ffff0000 broadcast 128.1.255.255
```


© 2003 Hewlett-Packard Development Company, L.P.

1–8. SLIDE: The IP Network Address



Student Notes

The last few slides have covered the basic concepts required to formulate and understand IP addresses. The next few slides discuss several special IP addresses that you will likely encounter. The first of these is the IP Network Address.

An **IP Network Address** is a special address used by routers and other network devices to reference an entire network of hosts. The network address is formulated by setting all of the host bits in an IP address to "0."

Consider the examples on the slide. In the 128.1.x.x/16 IP addresses, the last 16 bits (that is, the bits in the last two octets) define the host portion of the addresses. Setting these 16 bits to "0" yields the following network address:

```
10000000.00000001.00000000.00000000 = 128.1.0.0/16
```

In the 192.1.1.x/24 IP addresses, the last 8 bits (that is, the bits in the last octet) define the host portion of the addresses. Setting these bits to "0" yields the following network address:

```
11000000.00000001.00000001.00000000 = 192.1.1.0/24
```


© 2003 Hewlett-Packard Development Company, L.P.

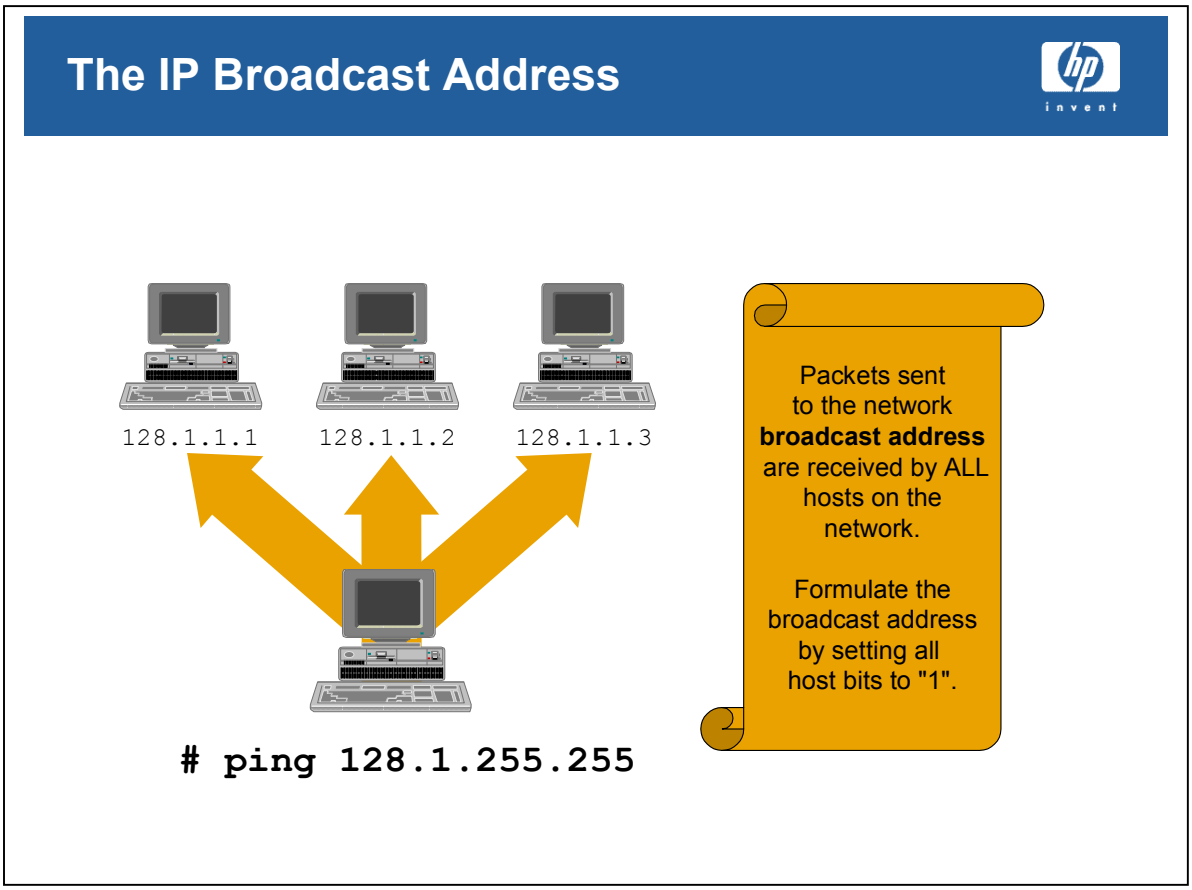
Viewing the Network Address

HP-UX systems automatically compute their network addresses by doing a binary "AND" operation on the IP address and IP netmask during system startup. You can view your system's network addresses using the **`netstat`** command:

```
# netstat -in
Name    Mtu  Network     Address     Ipkts   Opkts
lan0    1500 128.1.0.0   128.1.1.1   55670   23469
lo0     4136 127.0.0.0   127.0.0.1    3068    3068
```


© 2003 Hewlett-Packard Development Company, L.P.

1–9. SLIDE: The IP Broadcast Address



Student Notes

The network broadcast address may be used to send a packet to all of the nodes on a host's network. Some network services take advantage of this broadcast functionality to enable clients to identify an available server. X-terminals, for instance, may use the broadcast mechanism to identify all available login servers on the terminal's network. Network Information Service clients use the broadcast address to identify an NIS domain server during system startup. These are just a few of the many network services that use an IP broadcast to send a packet to all hosts on a network.

To formulate the broadcast address, simply set all IP host bits to "1". Consider the example on the slide. The 128.1.0.0/16 network has 16 host bits in the last two octets. Placing a "1" in all 16 host bits yields the following broadcast:

```
10000000.00000001.11111111.11111111 = 128.1.255.255
```


© 2003 Hewlett-Packard Development Company, L.P.

Viewing the Broadcast Address

HP-UX systems automatically compute their broadcast addresses during system startup. You can view your system's network addresses using the **ifconfig** command. First, use the **lanscan** command to determine the "Interface Name" that has been assigned to each of your LAN cards:

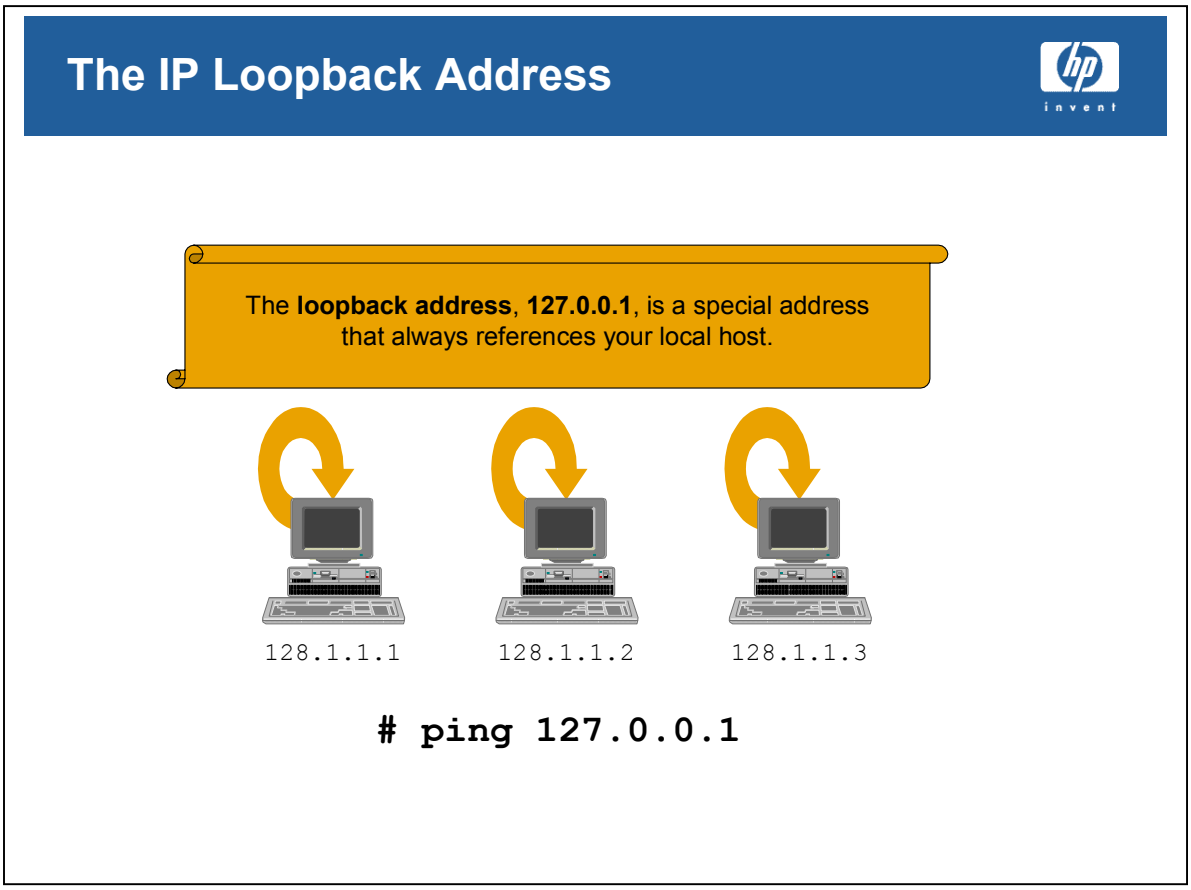
```
# lanscan
Hardware Station        Crd Hdw   Net-Interface  NM  MAC   HP-DLPI DLPI
Path     Address        In# State NamePPA        ID  Type  Support Mjr#
2/0/2    0x0800094A7334 0   UP    lan0 snap0     1   ETHER Yes     119
4/0/1    0x080009707AF2 1   UP    lan1 snap1     2   ETHER Yes     119
```

Next, use the **ifconfig** command to view each LAN card's broadcast address:

```
# ifconfig lan0
lan0: flags=843<Up,BROADCAST,RUNNING,MULTICAST>
      inet 128.1.1.1 netmask ffff0000 broadcast 128.1.255.255
```


© 2003 Hewlett-Packard Development Company, L.P.

1–10. SLIDE: The IP Loopback Address



Student Notes

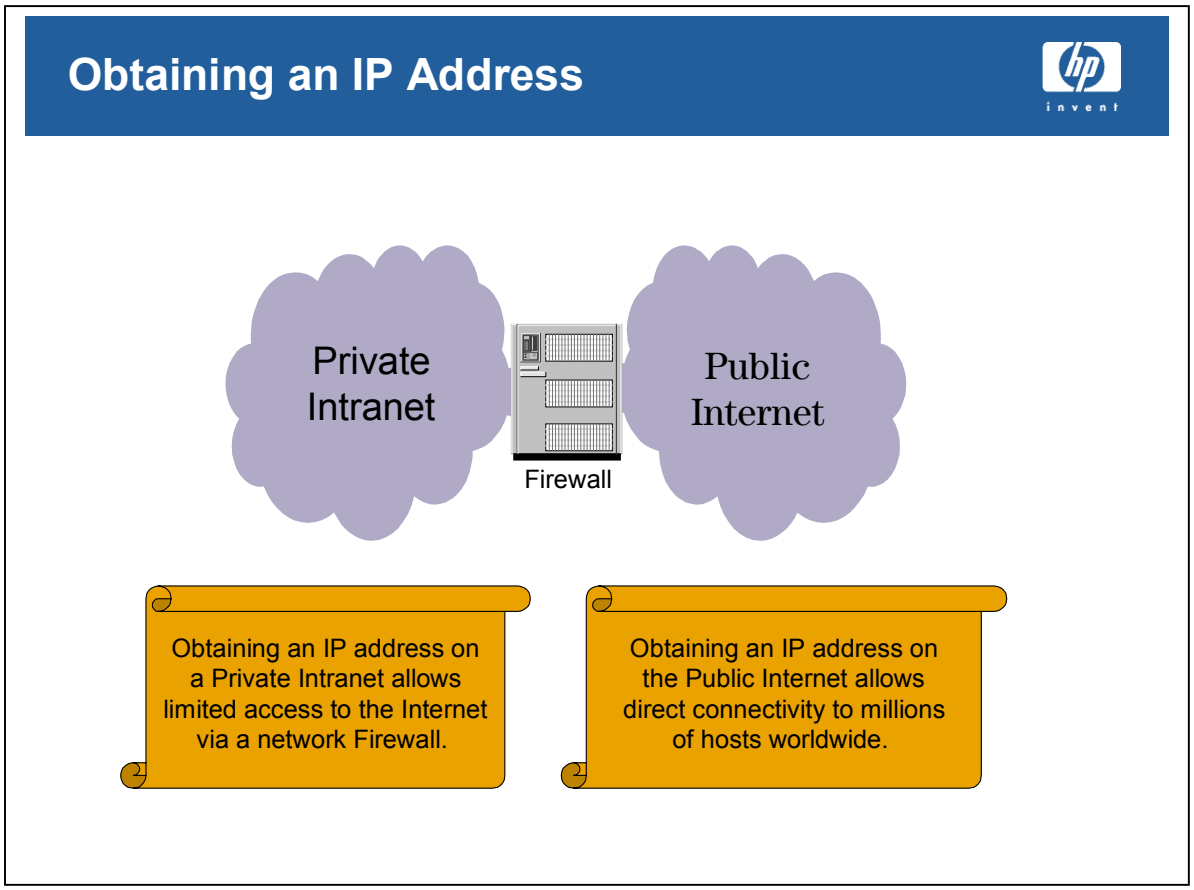
The IP **`loopback`** (or **`localhost`**) address is a special IP address that may be used to reference your local host, without actually sending a packet out on the local network. Applications sometimes use the **`loopback`** address to send network traffic to other processes on the same machine. The **`loopback`** address may be used for troubleshooting purposes as well. For instance, if a client claims to be having difficulty establishing a **`telnet`** connection to your host, **`telnet`** your **`loopback`** address. If *your* **`telnet`** attempt to the **`loopback`** address succeeds, there is probably a network connectivity problem between your host and the client, rather than a problem with the **`telnet`** service.

Attempts to access the **`loopback`** address should succeed even if your LAN card is down, disconnected, or misconfigured.

The **`loopback`** address is always set to 127.0.0.1.

© 2003 Hewlett-Packard Development Company, L.P.

1–11. SLIDE: Obtaining an IP Address



Student Notes

Every host on an IP network must have an IP address. The procedure required to obtain an IP address depends on the network you wish to connect to.

Connecting to the Public Internet

A direct connection to the **public Internet** allows direct connectivity to millions of hosts connected to the Internet worldwide. This offers great flexibility, but also some danger. Connecting directly to the public Internet also potentially allows hackers all over the world to access your host!

If you, or your organization, wish to have a direct Internet connection, you must obtain a unique IP address, used by no one else anywhere on the Internet. The International Committee for Assigned Names and Numbers (ICANN) is the organization that is currently responsible for determining how IP addresses are allocated and used. ICANN's website is accessible at **`http://www.icann.org`**. ICANN has delegated responsibility for allocating IP addresses out to several regional authorities:

`http://www.arin.net`	(North and South America)
`http://apnic.net`	(Asia and Pacific Region)
`http://ripe.net`	(Europe)

© 2003 Hewlett-Packard Development Company, L.P.

These organizations, in turn, allocate blocks of public Internet IP addresses to corporations and Internet Service Providers. Check with your local IT department or ISP to obtain an address on the public Internet.

Connecting to a private Intranet with an Internet Address

Many organizations choose not to connect individual hosts directly to the public Internet for security reasons. Why expose your hosts to thousands of hackers, if those hosts need only limited access to the outside networks?

Instead, many organizations choose to configure a **private Intranet** that is insulated from the dangers of the public Internet by some sort of network firewall. Firewalls can be used to control the type of traffic that passes both in and out of your organization's private Intranet.

There are two ways to obtain and allocate IP addresses in this situation. One approach is to request a public Internet IP address for each host, then shield those hosts behind your firewall. If you choose to go this route, you will have to apply for a block of unique, public Internet addresses from your ISP or the websites listed in the previous section.

Connecting to a private Intranet Using Network Address Translation

Since public Internet IP addresses are in short supply, many organizations choose instead to provide Internet access to their internal hosts using some sort of **proxy server** software, which does not require a unique Internet address for every host on the private Intranet. Using this approach, hosts on your private Intranet are assigned addresses from the following blocks of IPs:

```
10.*.*.*
172.16-31.*.*
192.168.*.*
```

These addresses are designated specifically for use on private Intranets. Hosts with addresses within these ranges may not be connected directly to the public Internet, nor are packets destined for these addresses allowed to pass on or through the public Internet. Since these addresses are not allowed directly on the public Internet, any organization may use these addresses without fear of conflicting with other organization's addresses.

Question: If packets destined for these addresses are not allowed on the public Internet, how can these hosts send email or access web sites outside their private networks?

Intranet hosts that need web access to the outside world may access the Internet via a **proxy server**. These hosts can be configured to relay all external web access requests through a specially configured server with connections both to the private Intranet, and the public Internet. The proxy server forwards internal clients' access requests to external sites via its IP address on the public Internet, then relays the responses back to the requesting clients.

Email service may be provided using similar functionality. Hosts on the private Intranet send and receive email via a specially configured **Mail Gateway** that straddles both the private Intranet, and the public Internet.

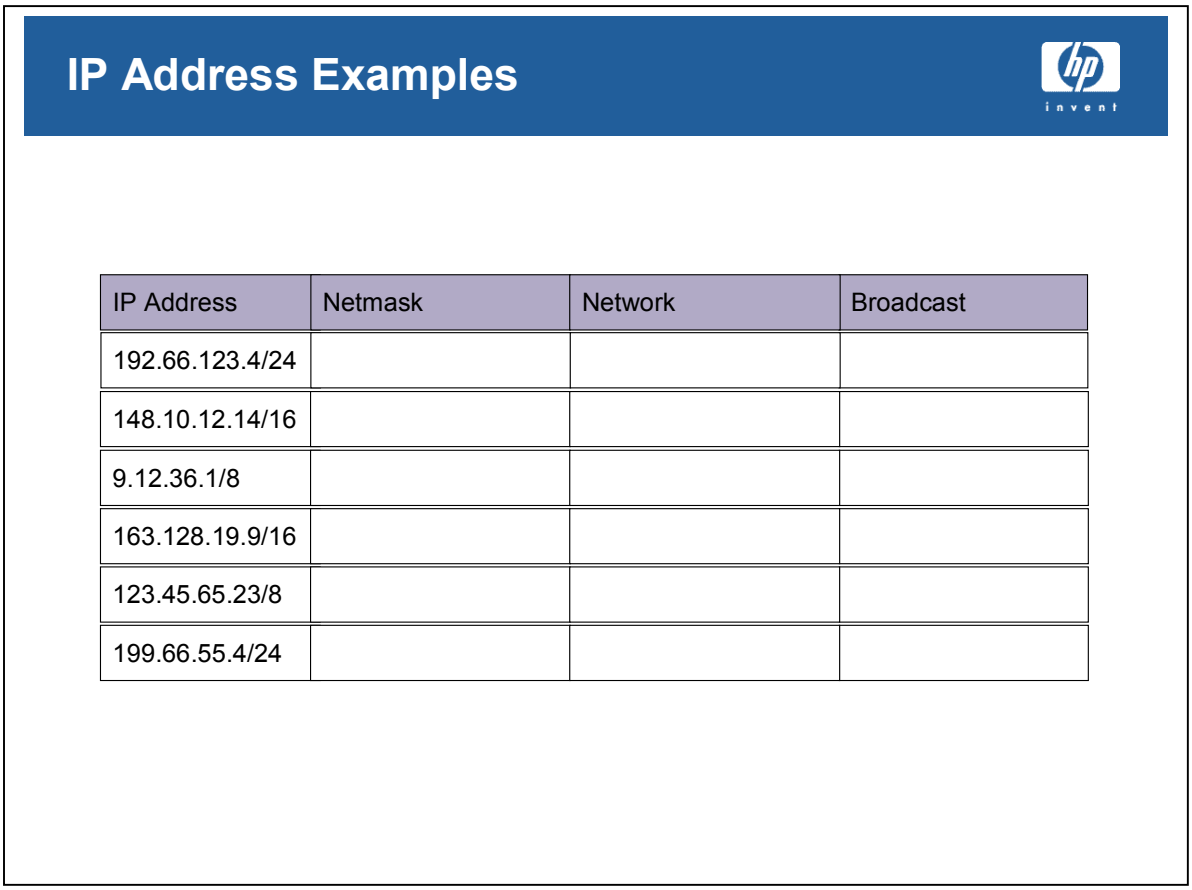
For even more flexibility, many firewall packages can be configured to provide **Network Address Translation** service. Using this functionality, clients on the private Intranet can

© 2003 Hewlett-Packard Development Company, L.P.

relay requests for many different network services through the corporate firewall. HP's Praesidium product is one of many products designed to provide this type of functionality.

© 2003 Hewlett-Packard Development Company, L.P.

1–12. SLIDE: IP Address Examples



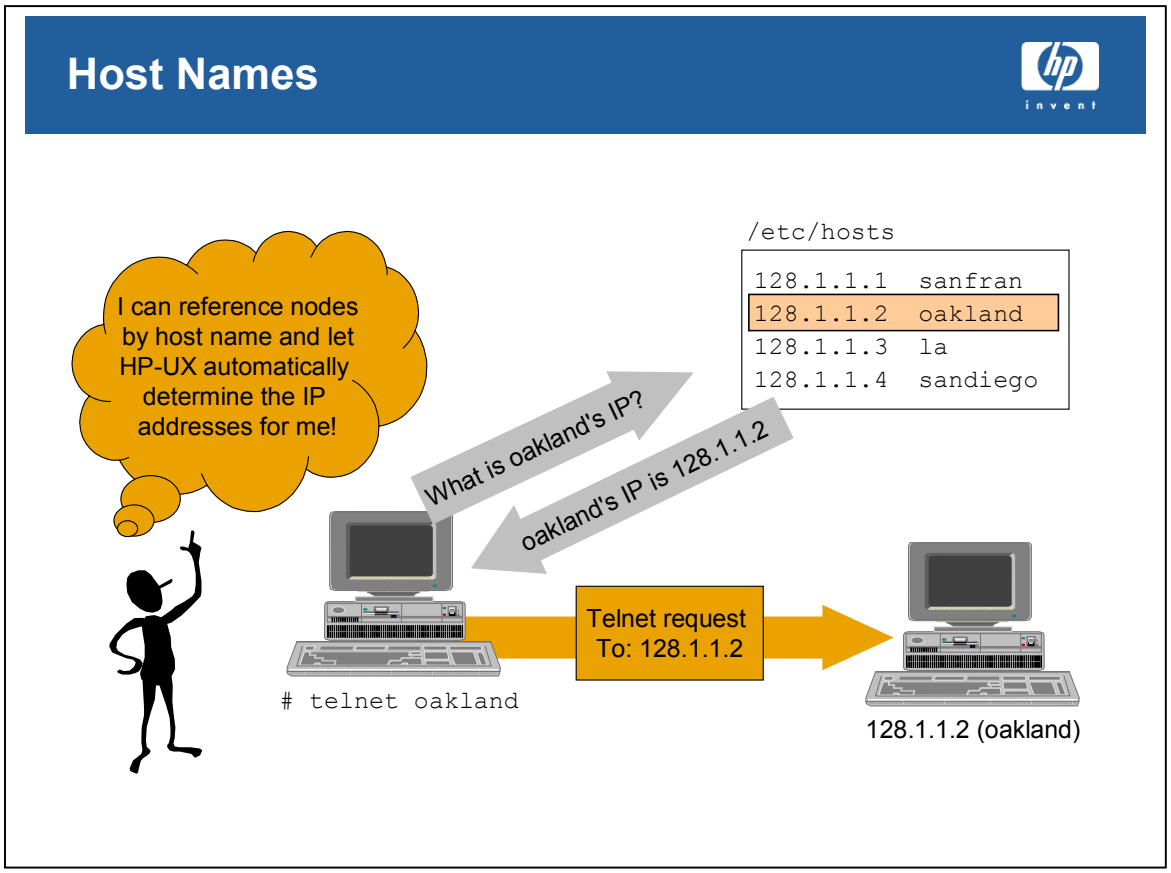
IP Address	Netmask	Network	Broadcast
192.66.123.4/24			
148.10.12.14/16			
9.12.36.1/8			
163.128.19.9/16			
123.45.65.23/8			
199.66.55.4/24			

Student Notes

The slide above lists six IP addresses in dotted decimal, "/" notation. Using the information given, compute the netmask, network, and broadcast address associated with each IP address.

© 2003 Hewlett-Packard Development Company, L.P.

1–13. SLIDE: Host Names



Student Notes

Although HP-UX systems and other network devices identify hosts by IP address, users and applications find IP addresses to be a cumbersome method for identifying network hosts:

- IP addresses are not very memorable. Users that access dozens of network hosts on a regular basis may have trouble remembering those hosts' IP addresses.
- Anytime you change your network topology, IP addresses are likely to change. Updating all the scripts and application configuration files that reference the old IP addresses could quickly become a support nightmare!

For both of these reasons, many users and applications prefer to reference network hosts by host name rather than IP address. A **host name** is nothing more than a user-friendly, easily remembered, "nickname" assigned to each host on a network.

Choosing Host Names

There are just four rules to remember when choosing system host names:

- The maximum length for a host name is eight characters.
- Host names must only contain letters, numbers, and underscores. Punctuation marks and other special characters are not allowed.

© 2003 Hewlett-Packard Development Company, L.P.

- Every host name must be unique.

- Choose meaningful host names. A system's host name may be based on the primary user (the workstation on Tom's desk might have host name "tom"), function ("mailsvr" or "filesvr"), geography ("chicago", "tokyo"), or any other scheme that your users find memorable.

Resolving Host Names to IP Addresses

Although users may prefer to identify hosts by host name, every host must still have an IP address, and every outgoing packet must have a destination IP address. Somehow, the host names specified by your users must be resolved to IP addresses recognized by your network devices. There are three mechanisms available for converting host names to their corresponding IP addresses.

The **`/etc/hosts`** file	Each system maintains its own file which lists the names and IP addresses of other nodes on the network. This is used primarily on small networks.
NIS	One system (the NIS server) maintains a list of all the nodes and IP addresses on the network. When resolving host names to IP addresses, all systems reference the NIS server. This is used on medium size networks.
DNS	DNS uses a distributed database of host name/IP information. Thousands of DNS servers scattered across the Internet share responsibility for resolving host names to IP addresses, and share IP/host name resolution information back and forth as necessary. DNS is the host name resolution method of choice for large networks, and for hosts connected to the public Internet.

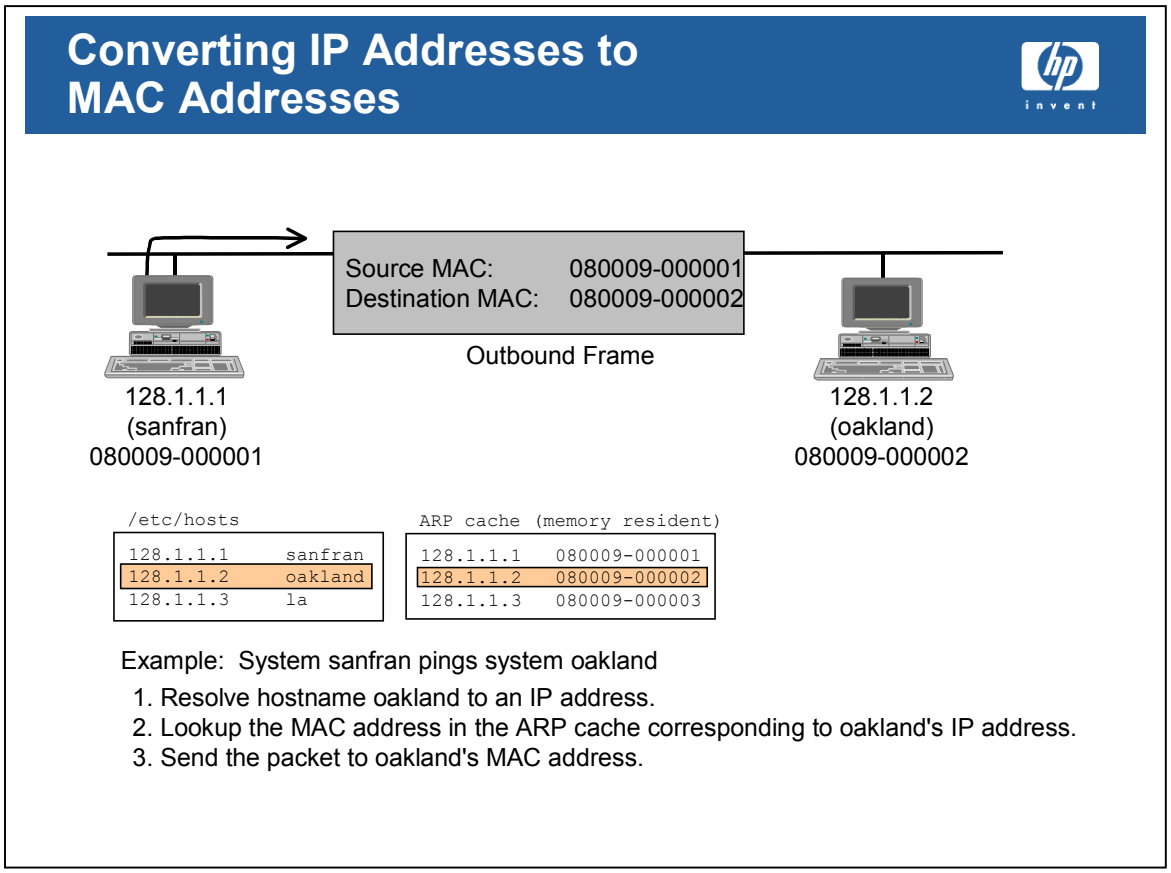
Viewing your Host Name

Use the **`hostname`** command to view your system host name.

```
# hostname
  sanfran
```


© 2003 Hewlett-Packard Development Company, L.P.

1–14. SLIDE: Converting IP Addresses to MAC Addresses



Student Notes

As you may recall from an earlier discussion of MAC addresses, every frame of data passed across a network must include both source and destination MAC addresses.

To allow the system to quickly determine a remote node's MAC address, each local kernel maintains a real-time, lookup table known as the **ARP cache**. The ARP cache maps IP addresses of remote nodes to their corresponding MAC addresses.

The Address Resolution Protocol (ARP) cache is a memory resident data structure whose content is maintained and managed by the local system's kernel. By default, the ARP cache contains the IP addresses and corresponding MAC addresses of nodes that the local system has communicated with in the last five minutes.

Explanation of the Slide Example

The slide above illustrates the lookup process a system uses when communicating with another node on the network. When system sanfran pings oakland, sanfran must first resolve oakland's host name to an IP address using the **`/etc/hosts`** file.

Next, sanfran checks the ARP cache to find the MAC address that corresponds to oakland's IP address.

© 2003 Hewlett-Packard Development Company, L.P.

Finally, sanfran can send the outbound frame on the network using oakland's MAC address as the destination.

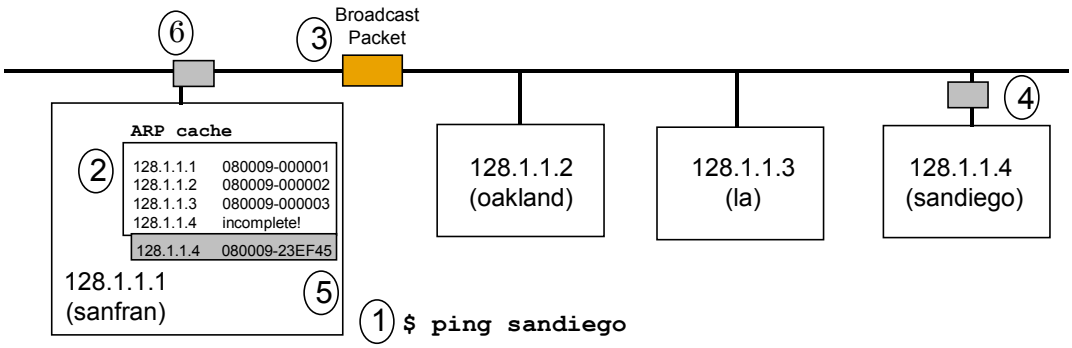
Viewing the ARP Cache

You may view the contents of your ARP cache at any time by issuing the **arp** command.

```
# arp -a
```


© 2003 Hewlett-Packard Development Company, L.P.

1–15. SLIDE: Populating the ARP Cache



Example: sanfran pings sandiego

1. sanfran pings sandiego. sanfran resolves sandiego's IP address via `/etc/hosts`.
2. Search for sandiego's IP in the arp cache — the IP address is not found in ARP cache.
3. Send ARP broadcast on the local network to find the MAC address for 128.1.1.4.
4. System with the specified IP address responds with a packet containing its MAC.
5. The MAC address and corresponding IP address are added to sanfran's ARP cache.
6. The frame specifically addressed to sandiego's MAC address is sent.

Student Notes

Resolving a destination node's IP address to its corresponding MAC address is fairly straightforward as long as the destination node's MAC address is in the local node's ARP cache. There are many situations however, when a destination node's MAC address may not be in the local ARP cache. What happens then?

How Does HP-UX Populate the ARP Cache?

If a local host cannot find a destination host's MAC address in the ARP cache, the local host does the following:

- The local host sends out a broadcast packet to all nodes on the network asking if their IP address matches the IP address in question.
- One and only one node should respond to the ARP broadcast by sending a reply packet indicating that it has the requested IP address. The reply packet sent by the remote node will contain the remote node's MAC address.
- Upon receiving the reply packet, the local node records the remote node's IP/MAC address information in the local ARP cache.

© 2003 Hewlett-Packard Development Company, L.P.

Explanation of the Slide Example

1. A user on sanfran attempts to **`ping`** sandiego.

   ```
   # ping sandiego
   ```

2. sanfran uses the **`/etc/hosts`** file to resolve "sandiego" to IP address 128.1.1.4.

3. Once sanfran determines sandiego's IP address, sanfran checks the ARP cache for sandiego's IP address. In this example, sandiego's IP address is not present in sanfran's ARP cache.

4. In order to determine sandiego's MAC address, sanfran sends an ARP broadcast onto the network requesting a response from the host with IP address 128.1.1.4 (sandiego's IP).

5. sandiego responds to sanfran's broadcast.

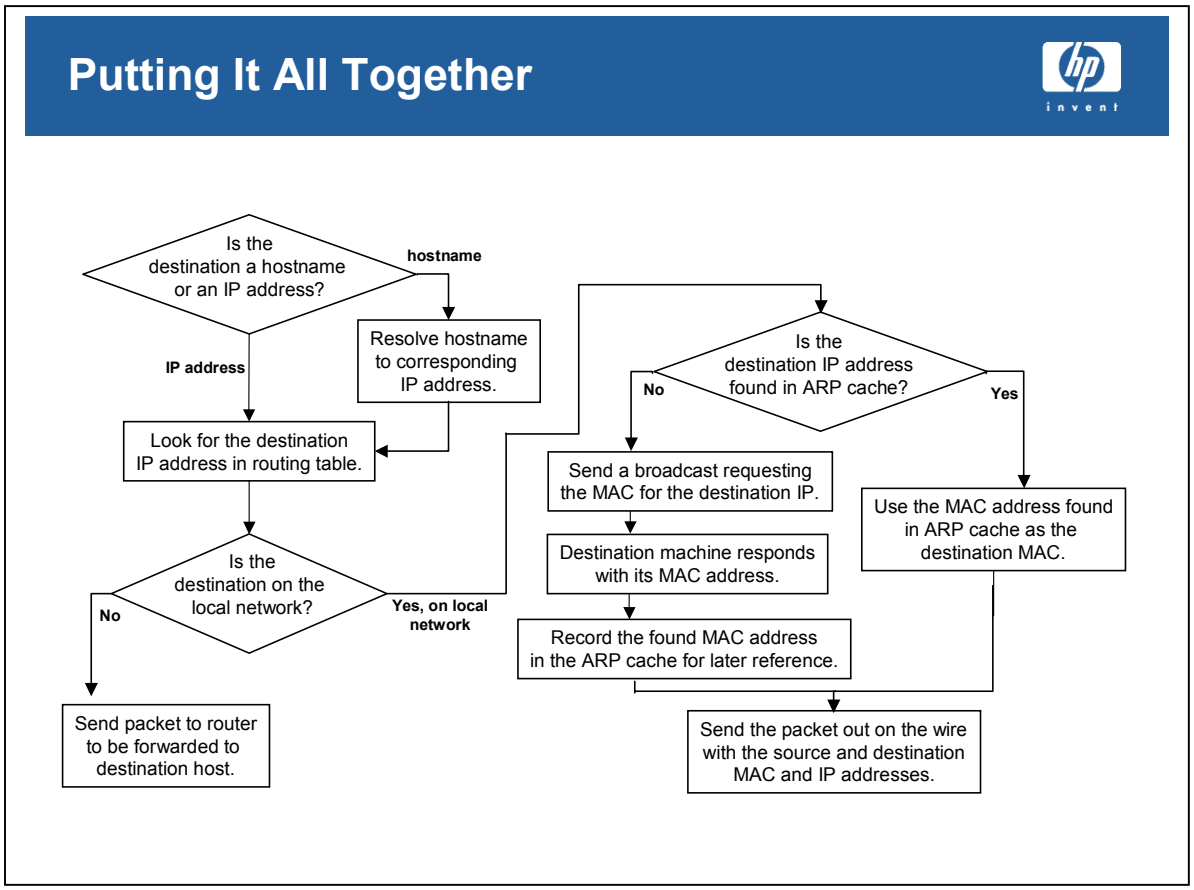
6. After receiving sandiego's response, sanfran adds sandiego's MAC address to the local ARP cache for future reference.

7. sanfran can now **`ping`** sandiego, addressing the packets specifically to sandiego's MAC address.

   ```
   #=> ping sandiego
   PING sandiego: 64 byte packets
   64 bytes from 128.1.1.4: icmp_seq=0. time=18. ms
   64 bytes from 128.1.1.4: icmp_seq=1. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=2. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=3. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=4. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=5. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=6. time=2. ms
   64 bytes from 128.1.1.4: icmp_seq=7. time=2. ms

   ----sandiego PING Statistics----
   8 packets transmitted, 8 packets received, 0% packet loss
   round-trip (ms)  min/avg/max = 2/4/18
   ```


© 2003 Hewlett-Packard Development Company, L.P.

1–16. SLIDE: Putting It All Together



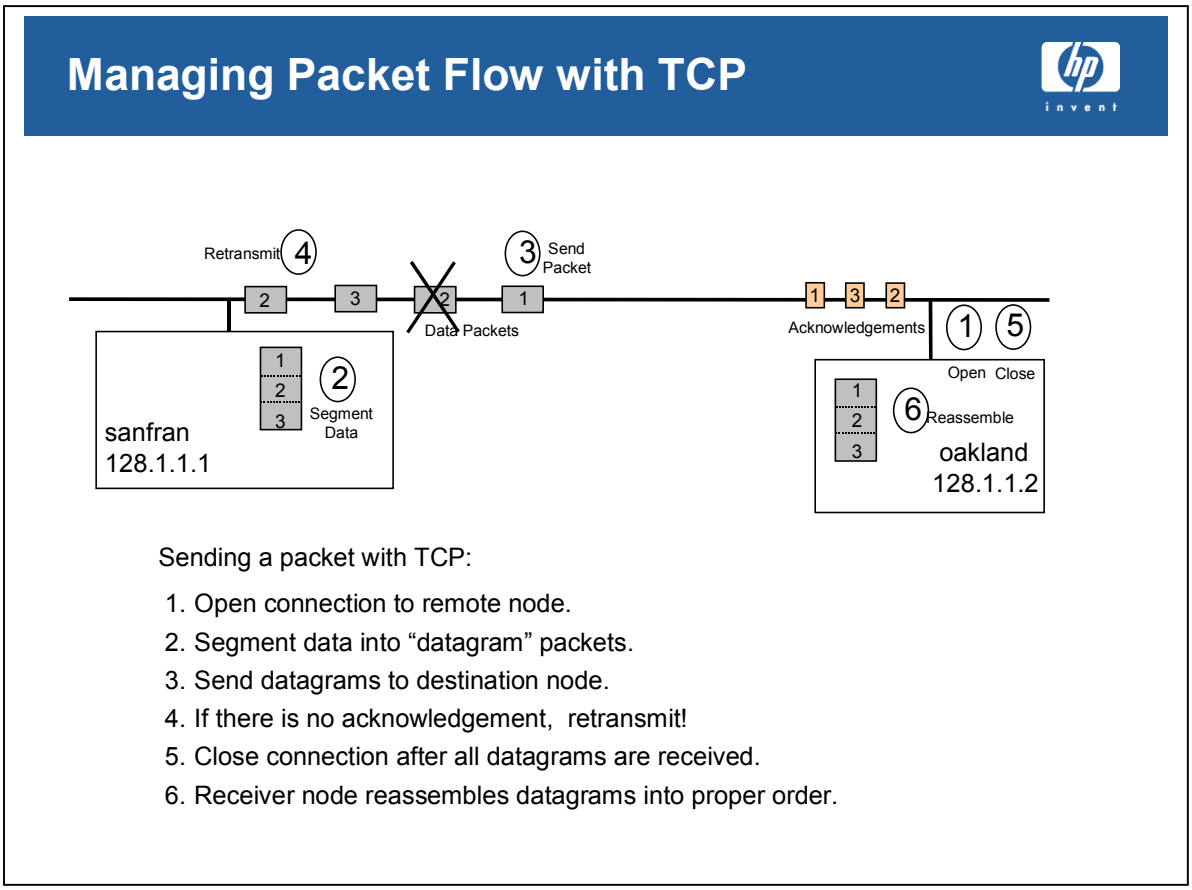
Student Notes

The flow chart above summarizes the actions that have to occur every time hosts communicate across a local area network.

The flowchart notes that packets sent to hosts outside of the local network must be forwarded to a router, before being passed to their eventual destination. Routing will be discussed in detail later in the course.

© 2003 Hewlett-Packard Development Company, L.P.

1–17. SLIDE: Managing Packet Flow with TCP



Student Notes

Up to this point, we have discussed how:

- Host names are resolved to IP addresses.
- How IP addresses are resolved to MAC addresses.

Several issues have not been addressed, yet, though:

- What happens when a packet arrives at the destination host? How is the packet passed to the destination application on that host?
- What happens if a packet is lost? Who is responsible for re-sending the lost packet or otherwise handling this situation?

The remaining slides in the chapter discuss two protocols that govern how packets are sent and acknowledged, and the port and socket addresses that ensure that data sent across a network is passed to the appropriate process or application on the destination host.

© 2003 Hewlett-Packard Development Company, L.P.

Transmission Control Protocol — TCP

The two main sets of rules governing how nodes communicate with each other are the TCP protocol and the UDP protocol. The TCP protocol requires more overhead, but provides more reliability than UDP.

Two important concepts characterize the TCP protocol.

TCP is a **Connection Oriented** protocol. A communication session is established between the two nodes before any data is exchanged.

TCP is a **Reliable** protocol. For every datagram sent, an acknowledgment is returned by the receiver. If an acknowledgment is not received, the transmitting node resends the packet.

Explanation of the Slide Example

The slide illustrates how data is transferred from one node to another using the TCP protocol.

1. Before any data is transferred, a communication session is established between the two nodes.

2. Before sending the data, the sending node segments the data into smaller datagram packets.

3. The datagram packets are sent to the destination node.

4. Upon receiving the datagram packets, the destination node sends acknowledgment packets back to the source node. The sending node automatically retransmits unacknowledged datagrams.

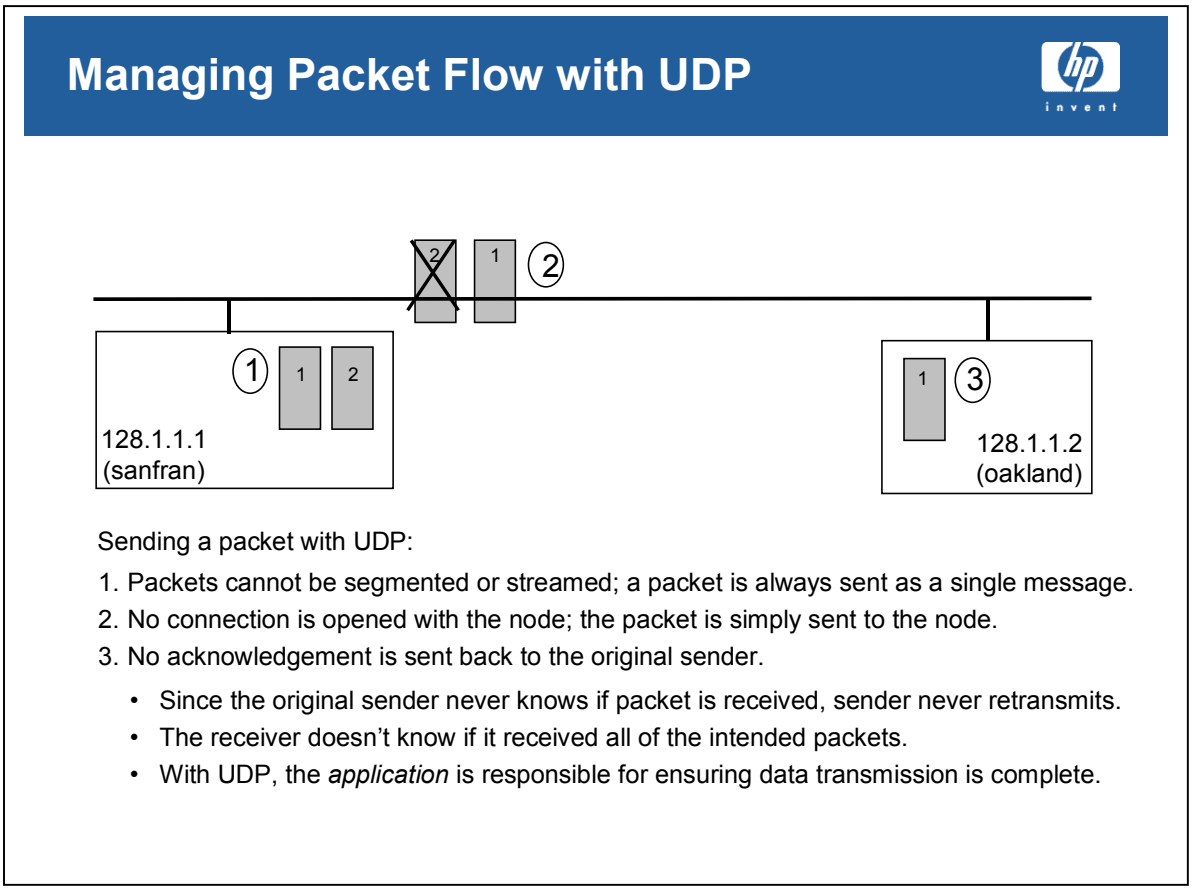
5. Upon successfully transferring all datagrams to the destination node, the connection between the two nodes is terminated and closed.

6. Once the destination node has received all datagrams, they are reassembled in their proper sequence.

NOTE:	In some cases, steps 5 and 6 may occur in reverse order.

© 2003 Hewlett-Packard Development Company, L.P.

1–18. SLIDE: Managing Packet Flow with UDP



Student Notes

The second common protocol used between two nodes on a network is the User Datagram Protocol (UDP). UDP requires less network overhead than TCP, but it does not provide an acknowledgement mechanism. It is therefore considered unreliable. Characteristics of the UDP protocol are below.

UDP is a **Connectionless** protocol. No communication session is established before the source node sends the first datagram.

UDP is an **Unreliable** protocol. The receiving node does not send acknowledgment packets back to the source node. The source node never knows whether the data packet arrived at the destination node. For this reason, the protocol is considered unreliable.

Explanation of the Slide Example

The slide shows an example of two datagrams being sent using the UDP protocol.

1. sanfran wants to send data to host oakland. The data is not segmented or fragmented; rather, it is sent as a single datagram (max size 64 KB).

2. No connection is established with the destination node. The datagram is simply sent to the destination address.

© 2003 Hewlett-Packard Development Company, L.P.

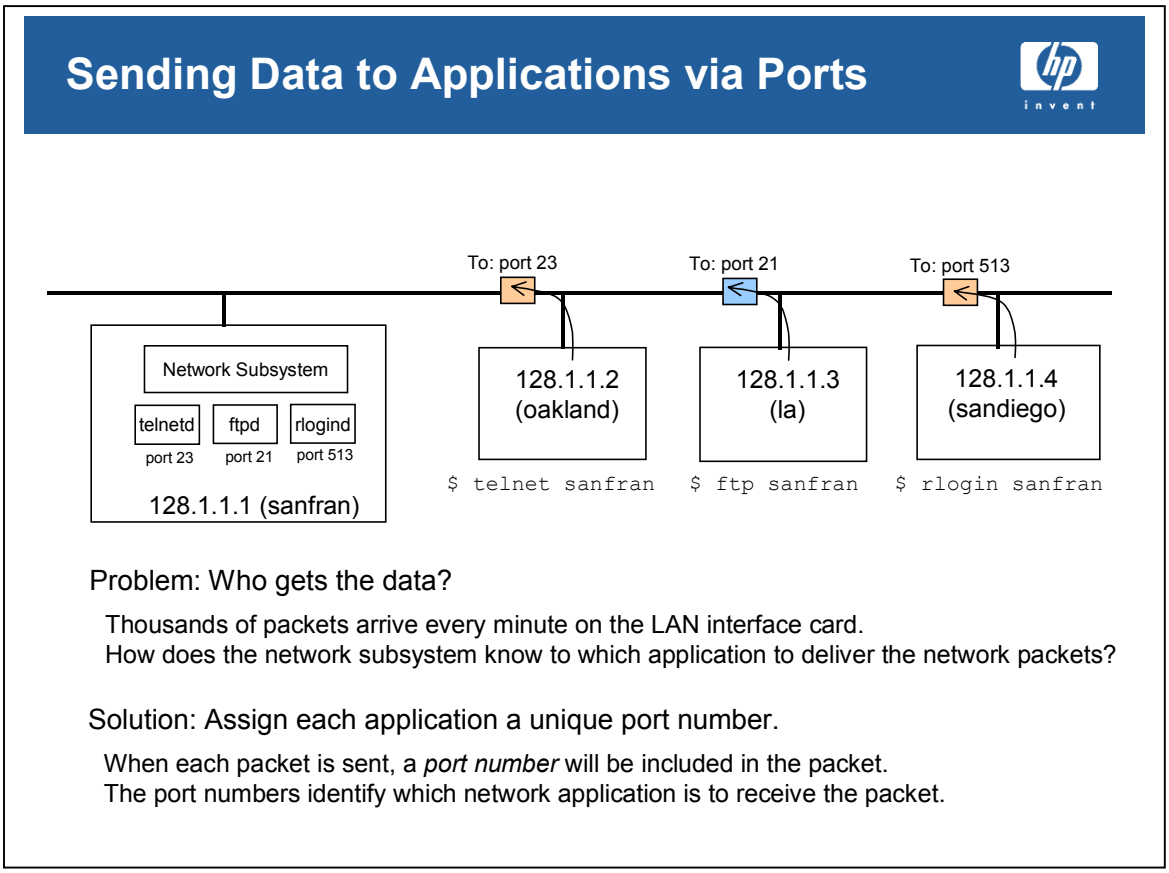
3. UDP does not send an acknowledgement back to the sender. Acknowledgement, if desired, must be handled by the application, not by the underlying UDP protocol.

Analogy: Sending data via UDP is similar to mailing a letter through the postal service. No connection between the sender and receiver is established before the letter is sent, nor is any acknowledgement returned after the letter is received.

Analogy: Sending data via TCP is similar to making a phone call. Before any communications takes place, a connection is established between the sender and receiver. There is a verbal acknowledgment that information is being received.

© 2003 Hewlett-Packard Development Company, L.P.

1–19. SLIDE: Sending Data to Applications via Ports



Student Notes

MAC addresses, IP addresses, TCP and UDP are all used to get packets from node to node on a network. Each node, though, may have dozens, if not hundreds, of network services and applications running simultaneously. When a data packet arrives on a system's LAN interface, how does HP-UX determine which application should receive that packet?

Port Numbers

Every network application is assigned a unique **port number** that distinguishes that application from all others. Network hosts specify which application should receive a packet by including a destination port number in outgoing packets.

Explanation of the Slide Example

The example on the slide shows three client systems. Each client system is accessing a different network service on server sanfran. The clients identify the desired service by port number.

oakland's **`telnet`** request is destined for sanfran's **`telnetd`** process on port number 23. la's **`ftp`** request is destined for sanfran's **`ftpd`** process on port number 21. sandiego's **`rlogin`** request is destined for sanfran's **`rlogind`** daemon on port number 513.

© 2003 Hewlett-Packard Development Company, L.P.

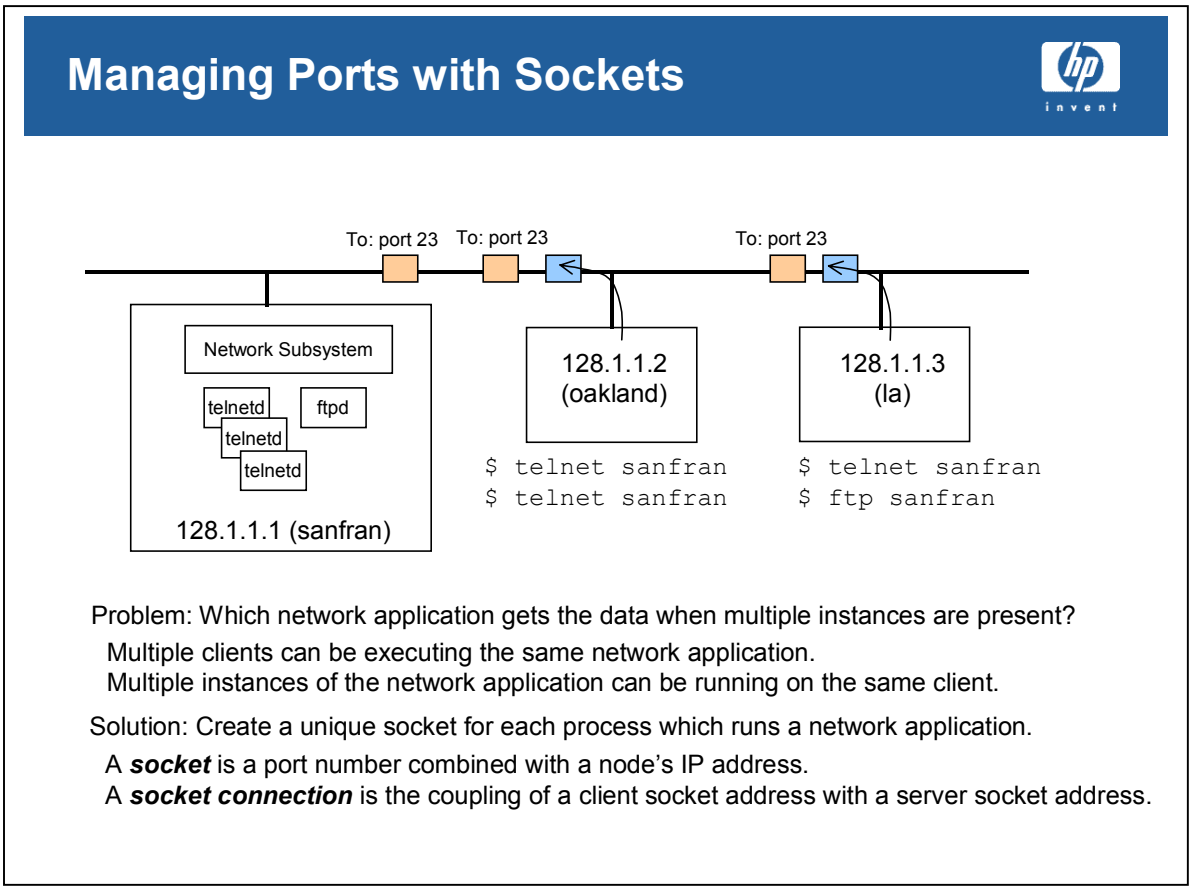
As the flood of incoming packets arrives, sanfran ensures that each packet gets to the right application or service by checking the destination port numbers.

The `/etc/services` File

In order for clients to be able to access the network services successfully, port numbers for network service server processes must be consistent. The most common network services use predefined port numbers that are consistent across all hosts. These well-known port numbers for the standard network applications and services are defined in the **`/etc/services`** file on all HP-UX (and most other UNIX) systems.

© 2003 Hewlett-Packard Development Company, L.P.

1–20. SLIDE: Managing Ports with Sockets



Student Notes

A packet's destination application can be identified by the packet's destination port number. What happens, though, if:

- Clients oakland and la both choose to access the **`telnet`** service on server sanfran simultaneously? Both nodes address their packets using port number 23, yet each packet must be handled by a separate instance of the **`telnetd`** daemon.

  How does sanfran distinguish between **`telnet`** packets from one node versus **`telnet`** packets from another node?

- User1 and user2 on oakland initiate simultaneous **`telnet`** sessions to sanfran. Both **`telnetd`** processes on sanfran use the well-known **`telnet`** port number, 23.

  How do sanfran and oakland determine which telnet packets belong to user1, and which belong to user2?

© 2003 Hewlett-Packard Development Company, L.P.

Sockets

Sockets provide the solution to both of the problems mentioned above. A socket is simply an address that identifies a specific network application running on a specific host. A socket address is formed by appending a destination port number to a destination IP address.

The sockets used by the applications on the slide are listed below:

128.1.1.1.23 The socket for the **`telnetd`** daemon on sanfran.

128.1.1.1.21 The socket for the **`ftpd`** daemon on sanfran.

128.1.1.2.50001 The socket for the first **`telnet`** program on oakland.

128.1.1.2.50002 The socket for the second **`telnet`** program on oakland.

128.1.1.3.50001 The socket for the **`telnet`** program on LA.

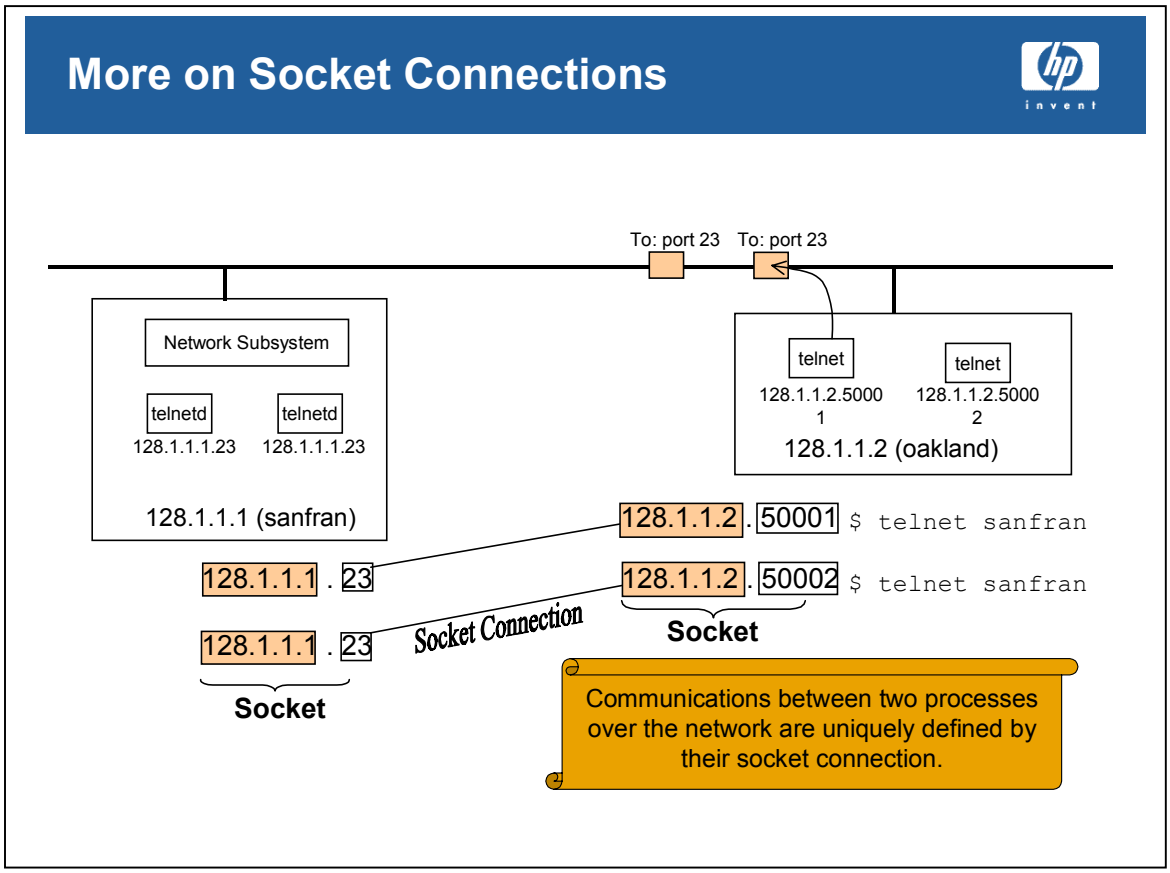
128.1.1.3.50002 The socket for the **`ftp`** program on LA.

Socket Connection

A socket connection is defined by the pairing of two sockets together. The first socket identifies a network program on a client node (*128.1.1.2.50001*), and the second socket identifies a network daemon (usually) on the server node (*128.1.1.1.23*). The socket connection would then be *128.1.1.2.50001–128.1.1.1.23*.

© 2003 Hewlett-Packard Development Company, L.P.

1–21. SLIDE: More on Socket Connections



Student Notes

The slide shows how sockets and socket connections can be used to uniquely identify two **`telnet`** service connections between client oakland and server sanfran.

When the first **`telnet`** instance is started on oakland, HP-UX assigns a port number for the **`telnet`** client process. Since there is no pre-defined port number for the client side **`telnet`** program, the first available port number is chosen (port number 50001 in the example on the slide). Thus, the socket created for the first telnet instance on oakland is 128.1.1.2.50001.

Oakland initiates a connection request to sanfran's well-known **`telnetd`** port, 23. Sanfran spawns a **`telnetd`** daemon to service the telnet request from oakland. This **`telnetd`** daemon uses port number 23. Therefore, the socket created to represent the **`telnetd`** daemon is 128.1.1.1.23.

The socket connection representing this communication session is 128.1.1.2.50001-128.1.1.1.23.

The second **`telnet`** session shown on the slide is using socket addresses 128.1.1.2.50002-128.1.1.1.23.

© 2003 Hewlett-Packard Development Company, L.P.

Thus, each of these connections may be uniquely identified by the pairing of the server and client processes' socket addresses.

© 2003 Hewlett-Packard Development Company, L.P.

1–22. SLIDE: Revisiting the OSI Model

Revisiting the OSI Model

7	Application	Creates/receives the data.
6	Presentation	Determines the format in which to represent the data. Possible choices are EBCDIC or ASCII format.
5	Session	Establishes a unique communication path between client/server. **Sockets** are used to communicate between two systems. A socket is an IP address plus a **port** number.
4	Transport	**TCP** requires that a socket connection be established; **UDP** does not. TCP requires packets be acknowledged; UDP does not. TCP is streams-based; UDP is message-based.
3	Network	**IP addresses** define a system's network and host number.
2	Data link	**MAC** addresses uniquely identify a LAN card. Ultimately, packets are sent from one MAC address to another. ARP caches map IP addresses to MAC addresses.
1	Physical	The type of media used to connect the machines together. The type of cabling used for the network.

Student Notes

In this module, we have learned how

- Host names are resolved to IP addresses.
- IP addresses are converted to MAC addresses.
- TCP and UDP protocols are used to allow nodes to communicate on the network.
- Port numbers are used to identify network applications.
- Socket connections are used to uniquely identify a communication sessions between a network application on two different hosts.

Compare the notes you made to your OSI worksheet to the OSI model on the slide above.

© 2003 Hewlett-Packard Development Company, L.P.

1–23. REVIEW QUESTIONS: LAN Concepts and Components

Directions

Answer the following questions.

1. If a host has two LAN interface cards, will the MAC addresses of the two cards be the same, or different?

2. Is it possible to determine which network a host is on just by looking at the host's MAC address?

3. Complete the following table:

IP Address	Netmask	Network Address	Broadcast Address
167.12.132.5/16			
124.132.12.5/8			
213.1.231.45/24			

4. Which of the networks listed in question 3 would allow the fewest hosts? What is the maximum number of hosts allowed on that network?

© 2003 Hewlett-Packard Development Company, L.P.

5. How many different networks are represented by the list of IP addresses below?

```
132.1.1.3/16
132.2.1.1/16
132.1.1.2/16
132.1.1.1/16
132.1.2.1/16
132.1.2.2/16
```

6. What is the highest possible host IP address on the 158.153.0.0/16 network? What is the lowest possible host IP address on this network?

7. What is the difference between a destination port number and a destination IP address?

8. Name one major difference between UDP and TCP.

9. HP-UX provides three different methods for mapping host names to IP addresses. Name two.

© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 2 — LAN Hardware Overview

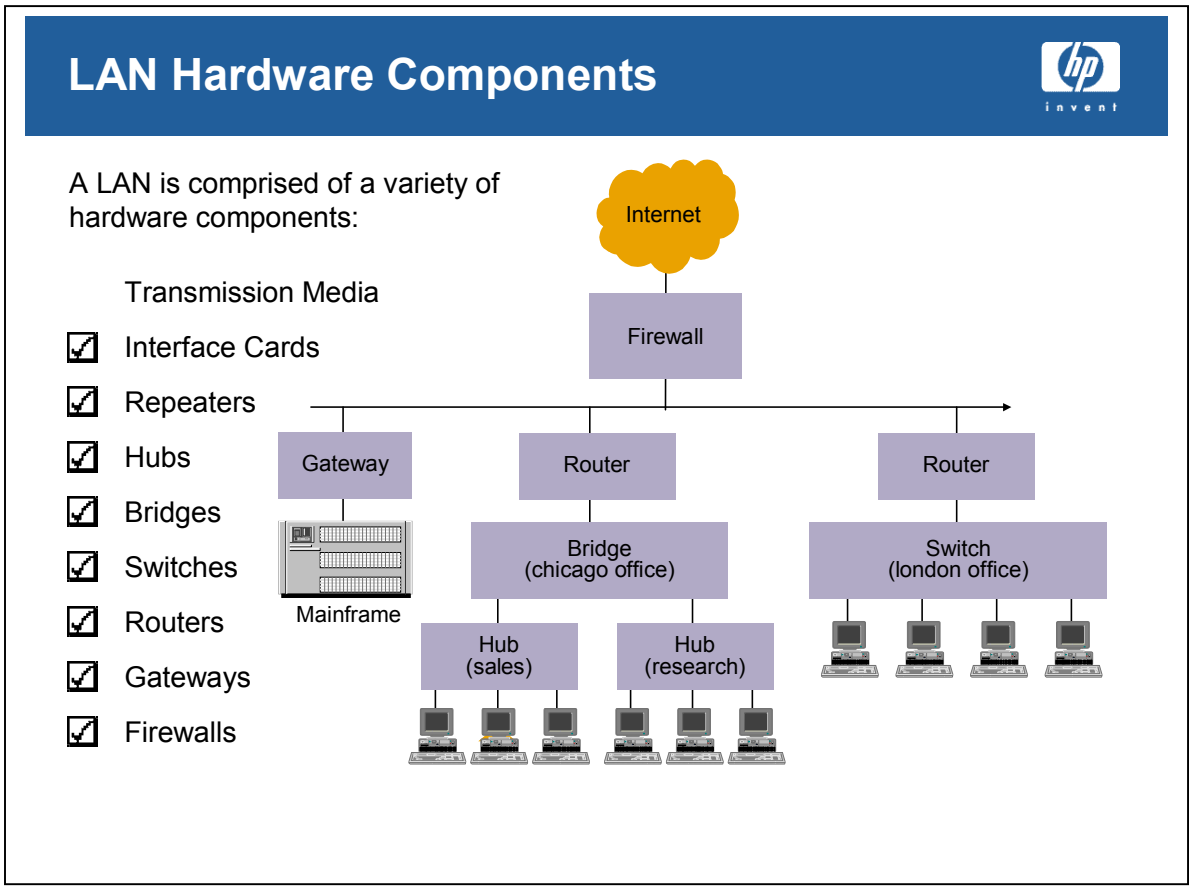
Objectives

Upon completion of this module, you will be able to do the following:

- Describe the characteristics of three major LAN cable types.
- Discuss three different LAN topologies.
- Explain two different LAN access methods.
- List the characteristics of an Ethernet LAN.
- List the characteristics of a Token Ring LAN.
- List the characteristics of an FDDI LAN.
- Explain the difference between physical and logical topologies.
- Describe the role of repeaters, hubs, bridges, switches, routers, gateways, and firewalls in a local area network.

© 2003 Hewlett-Packard Development Company, L.P.

2–1. SLIDE: LAN Hardware Components



Student Notes

Most LANs today are comprised of a variety of hardware components. Weeklong courses have been written about firewalls, routers, switches, and LAN topologies. Our goal in this chapter is simply to present an overview of the purpose and function of the most common hardware components you are likely to encounter as an HP-UX system administrator.

Every LAN usually has a combination of workstation and server nodes, each with one or more network interface cards (NICs). These nodes may be connected together via a variety of cable types in a variety of topologies. Different networking standards have different mechanisms for determining when hosts on the LAN are given the opportunity to transmit data. Most networks also include a variety of network devices. Some of the more common network devices include:

- repeaters
- hubs
- bridges
- switches

© 2003 Hewlett-Packard Development Company, L.P.

- routers
- firewalls

Each of these hardware components, devices, and topologies will be discussed in detail later in the chapter.

© 2003 Hewlett-Packard Development Company, L.P.

2–2. TEXT PAGE: OSI Worksheet

Table 1

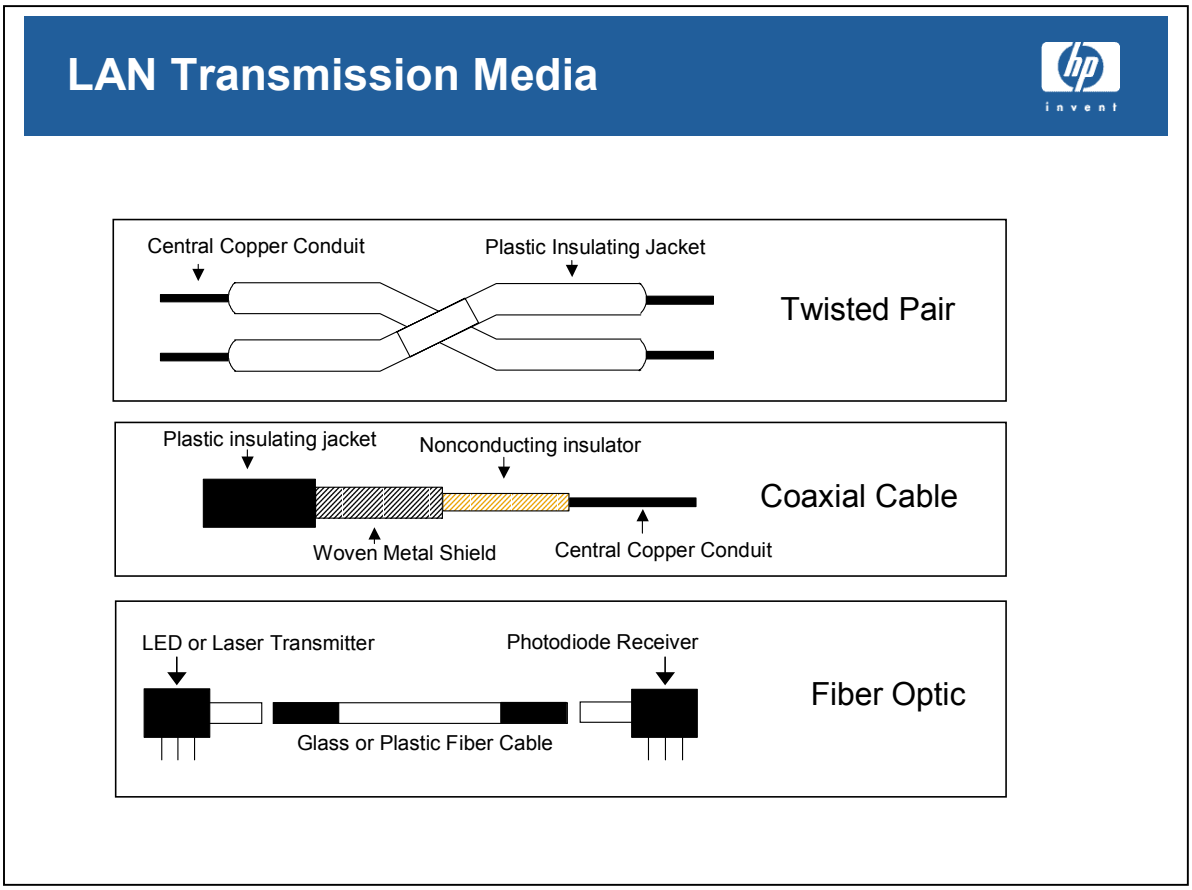
OSI Layer	Associated Protocols and LAN Hardware
7	
6	
5	
4	
3	
2	
1	

Instructions

During the lecture, a number of additional protocols and LAN hardware components will be discussed. Remove this sheet of paper from the workbook, and as your instructor introduces each new protocol and LAN hardware component, record it in the appropriate layer of the OSI chart.

© 2003 Hewlett-Packard Development Company, L.P.

2–3. SLIDE: LAN Transmission Media



Student Notes

Transmission media connects the devices in a local area network and provides the means by which data signals travel from device to device. Many different types of transmission media are used on today's networks. When choosing a transmission medium for your network, you must consider several issues:

- How much data must your network be able to handle? 10 Megabits per second (Mbps)? 100 Mbps? 1000 Mbps?

- Is electrical interference an issue in your environment? Some cable types are susceptible to data loss because of electrical interference from telephone lines, power cables, heavy electrical machinery, and fluorescent lights. This tends to be a more critical issue in manufacturing environments.

- What is the maximum distance between nodes on your network? Signals weaken as they travel along a cable. As the signals weaken, the effect of external electrical interference increases, and errors may occur. This signal loss is technically termed **attenuation**. Some transmission media types are more susceptible to attenuation than others.

- How much can you afford to spend? Some transmission media types are relatively cheap to purchase and install, while others are much more expensive.

© 2003 Hewlett-Packard Development Company, L.P.

The notes below describe some of the more common transmission media types used in today's networks.

Twisted-Pair Cable

Twisted-pair cable consists of two single wires, each encased in color-coded plastic insulation, and then twisted together to form a pair. Each pair of wires is then bundled with one to three other pairs, yielding a grand total of four or eight wires per cable. The cabling used to connect telephones is twisted-pair.

There are several variations on twisted-pair cable. Shielded Twisted-pair (STP) includes a foil or copper jacket to shield the wires inside the cable from electrical interference.

Unshielded Twisted-pair (UTP), which lacks shielding, is cheaper and much more common than STP in most networks today. Unshielded twisted-pair cable was originally designed for wiring telephones, but can be used for data as well. Since unshielded twisted-pair cable is already required in many buildings to support telephones, using this cable for your data needs as well can significantly reduce installation costs. UTP cable is available in several different grades:

Category 1 UTP:	Cat 1 UTP is used for doorbells, alarms, and other trivial applications; it is not appropriate for network applications.
Category 2 UTP:	Cat 2 UTP is primarily used for digital and analog phones; it is not appropriate for network applications.
Category 3 UTP:	Cat 3 UTP is used for 4 Mbps Token Ring, 10BaseT Ethernet, and analog and digital phone systems.
Category 4 UTP:	Cat 4 UTP is rare but sometimes used for 16 Mbps Token Ring networks.
Category 5 UTP:	Cat 5 UTP is used for 16 Mbps Token Ring, and 10BaseT, 100BaseT, and 1000BaseT Ethernet networks.
Category 5e UTP:	Enhanced Cat 5e UTP is a slightly higher-grade cable than standard Cat 5. Like Cat 5, Cat 5e can be used for Token Ring, 10BaseT, 100BaseT, and 1000BaseT Ethernet networks. Future network standards may require Cat 5e rather than Cat 5.

Standards are currently being developed for Cat 6 and Cat 7 cable grades that will support even higher data transmission rates in the future.

Cat 5 cable has been the cable of choice for most recent network installations. Cat 5e is an even better choice to ensure compatibility with future technologies. Twisted-pair cable is inexpensive, easy to install, and currently supports Token Ring and 10 Mbps through 1000 Mbps Ethernet networks.

Many purchased cables have "Cat 3," "Cat 5," or "Cat 5e" labels printed on the cables themselves so you can determine which type of cabling your shop uses. Cat 3, Cat 5 and Cat 5e twisted-pair cables all use standard 8-pin RJ-45 connectors that look very similar to standard telephone cables.

© 2003 Hewlett-Packard Development Company, L.P.

Coaxial Cable

Coaxial cable consists of a single, central conductive wire surrounded by a shield of either fine copper mesh or extruded aluminum. Between the shield and the center conductor is a dielectric (non-conducting) material. Cable TV boxes and cable modems both use variations on coaxial cable.

Two types of coaxial cable have been commonly used for LANs in the past:

Thicknet: (or ThickLAN) — Used a thick, inflexible coaxial cable. Adding a new node on a thicknet segment required the use of a "vampire tap." Tightening the vampire tap connector pierced the cable shielding and tapped into the cable's core. Because thicknet is so difficult to work with, it is very rarely used today.

Thinnet: (or ThinLAN) — Used a thinner, more flexible coaxial cable. Each thinnet cable has a "Bayonet-Neill-Concelman" (BNC) connector on each end. Nodes connect to a thinnet cable via a "T" shaped connector on the back of each node's network interface card. Every thinnet cable must be attached to a T-connector on both ends, and every open T-connector port must have a "BNC Terminator" to prevent loss of data. In order to add a node to a thinnet network, simply run a thinnet cable from an existing node's T-connector to the new node's T-connector, and connect a terminator if necessary.

Though thinnet coaxial cable is easy to install, it is more expensive than twisted-pair and does not support the newer 100BaseT and 1000BaseT network technologies. As a result, most new LAN installations use twisted-pair rather than coaxial cable.

Fiber-Optic Cable

Fiber-optic cable is made of glass or plastic fibers that transmit signals via light pulses. Fiber-optic cables can support extremely high data rates through a physically small cable. They are immune to electrical noise and are therefore able to provide a low error rate at a great transmission distance. The cable is inexpensive, but it is not easily tapped and is therefore difficult to install. Fiber-optic cable supports a transmission rate of 100 Mbps to 1000 Mbps. Fiber is often used for network backbones connecting multiple smaller department or workgroup LANs, since these applications may exceed the 100m segment limit imposed by twisted-pair. Fiber-optic is also commonly used in heavy industrial environments where interference poses problems for twisted-pair and for military applications where security is of paramount importance. There are two major categories of fiber-optic cable:

Multi-mode: Multi-mode fiber-optic cable typically has a 50 or 62.5-micron fiber-optic core surrounded by a 125-micron protective cladding (this is typically labeled 62.5/125 micron fiber-optic cable). Since multi-mode cable is relatively large, it is relatively easy to couple a light source to the cable. However, the larger core diameter allows the light to bounce off the sides of the cable, which leads to dispersion and signal degradation over distances greater than 2 km. LEDs are often used as the signal source on interface cards using multi-mode cable.

Single-mode: Single-mode fiber typically has a much smaller 10-micron core. This smaller core size minimizes dispersion and allows for much longer segment lengths — 100 km or more in some cases! The downside, however, is that single-mode fiber typically requires a relatively expensive laser, rather than an LED, as a signal source.

© 2003 Hewlett-Packard Development Company, L.P.

Most HP fiber-optic interface cards require 62.5/125 multi-mode cable with Straight Tip (ST), Subscriber Connect (SC), or Duplex SC type connectors. ST connectors are round in shape, while SC connectors are square; a Duplex SC connector is simply a pair of SC connectors in a single enclosure. Check your documentation to determine the specific cable/connector combination required for your environment.

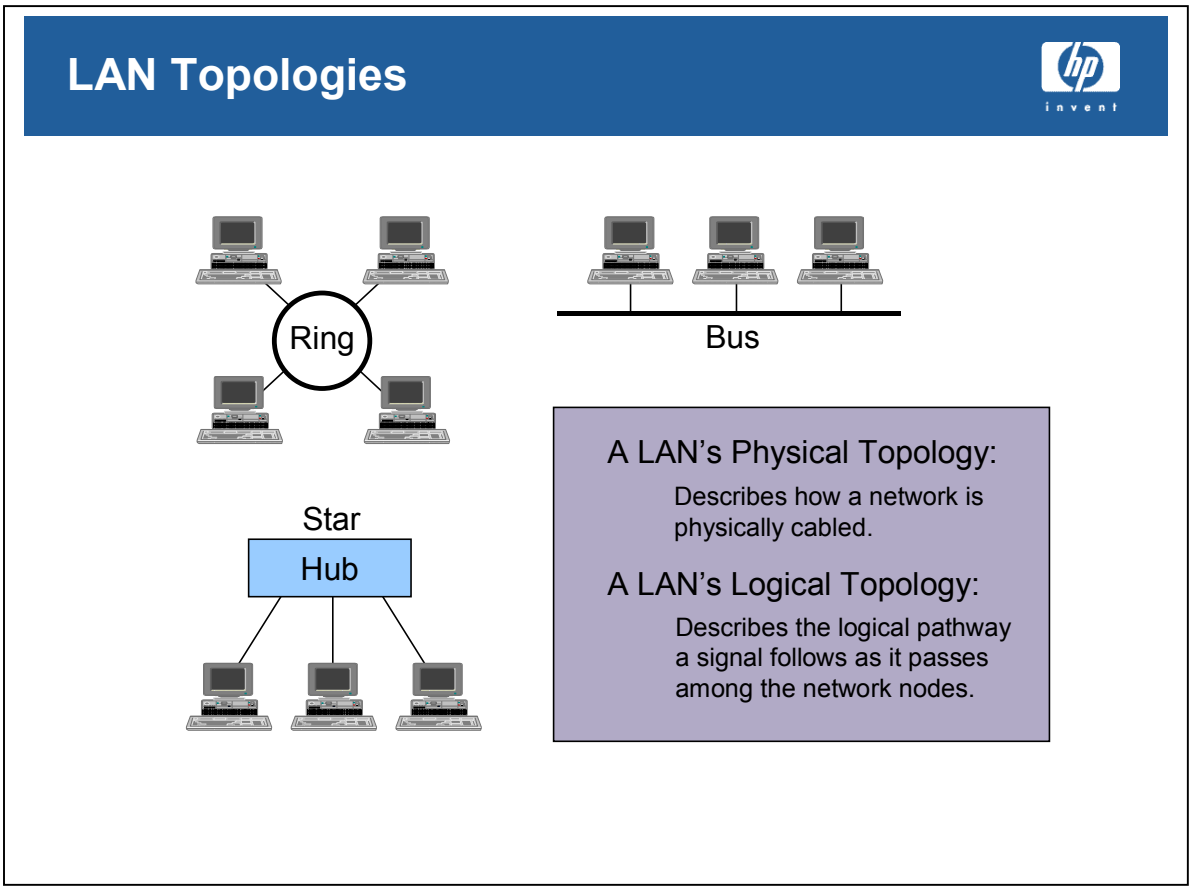
Comparison of LAN Transmission Media

Cable Type	UTP Twisted-pair	Coaxial	Fiber-optic
Connector Type	RJ-45 or 50 pin	BNC	Fiber-optic SC
Transmission Rate	10 Mbps to 1000 Mbps	10 Mbps	100 Mbps to 1000 Mbps
Maximum Segment	100m	185 m to 500 m	220 m to 1000 m+
Flexibility	Flexible	Stiff	Flexible
Noise Immunity	Good	Good	Excellent
Security	Moderate	Moderate	Excellent
Ease of Installation	Excellent	Good	Good
Cost per Connection	Very Low	Moderate	Expensive
Reliability	Good	Good	Excellent

* Adapted from HP's AdvanceStack Network Design Guide

© 2003 Hewlett-Packard Development Company, L.P.

2–4. SLIDE: LAN Topologies



Student Notes

Your LAN's **topology** determines the arrangement of the devices on your network. Three different topologies are commonly used today:

Bus Topology

Devices connected via a bus topology connect to a single, common, shared cable. Devices attach to the cable at regular intervals. Nodes attached to a network configured using a bus topology typically broadcast messages in both directions on the cable simultaneously. Ethernet standard networks usually use a bus topology when cabled via coaxial cable.

Ring Topology

Ring topology networks are cabled in a ring. Data is passed from node to node around the ring until it arrives at its destination. Some FDDI networks use a ring topology.

Star Topology

Star topology networks are the most common LAN type today. In a star topology network, cables radiate outward from a central device (typically called a **hub**) to each node on the network. Any time a host wishes to contact another host, it must send the signal to the hub,

© 2003 Hewlett-Packard Development Company, L.P.

which then propagates the signal to the desired destination. Ethernet networks using twisted-pair cable are cabled in a star topology.

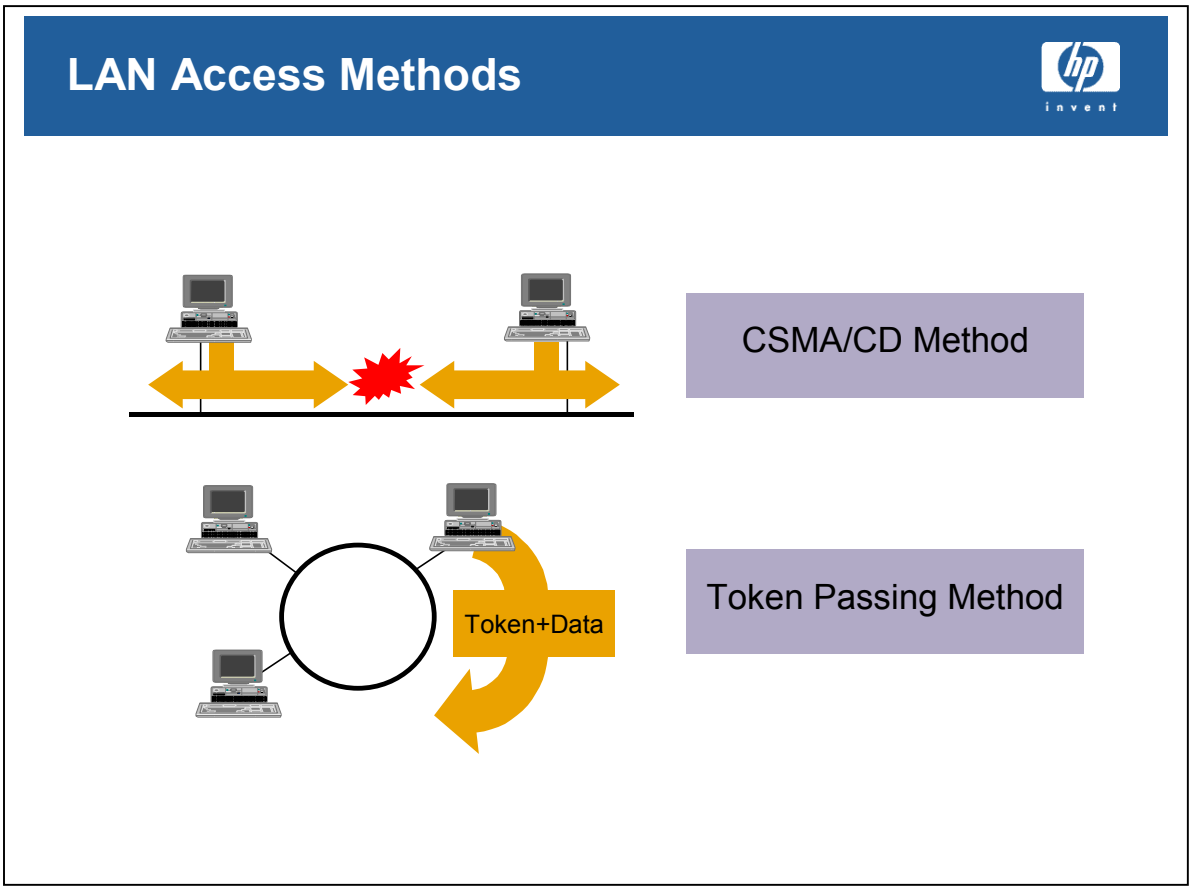
Physical versus Logical Topologies

A distinction should be drawn between the terms **logical topology** and **physical topology**. A network's **physical topology** determines how devices on the network are physically cabled. A network's **logical topology**, on the other hand, defines the logical pathway a signal follows from host to host.

In some cases, the physical topology may be identical to the logical topology, but in some cases, they may be different. For example, twisted-pair Ethernet networks use a *physical* star topology, but use a *logical* bus topology. Although cables radiate from a central Ethernet hub, the circuitry within the hub approximates the signal path of a bus topology network. Ethernet networks are not unique in this respect; Token ring networks are cabled using a star topology, but use a logical ring topology.

© 2003 Hewlett-Packard Development Company, L.P.

2–5. SLIDE: LAN Access Methods



Student Notes

After you have physically attached two or more nodes to your network, your network interface cards must determine *which* node is given an opportunity to transmit data and *when*. Several different **LAN access methods** have been used over the years to control access to local area networks. The two most common access methods are described below:

CSMA/CD — CSMA/CD stands for Carrier Sense Multiple Access with Collision Detection. Hosts on a CSMA/CD network monitor the network before transmitting. If a host has data to transmit, and the network is not already in use, the node transmits its signal on the wire. On a busy network, two nodes could potentially choose to transmit at the same time, resulting in a collision. If a collision occurs, the nodes responsible for the collision wait a random period, then retransmit. The random wait period makes it highly unlikely that the two nodes will retransmit at the same time again and create another collision. Ethernet networks use the CSMA/CD access method.

Token Passing — Hosts on LANs that use a token passing access method pass a "token" from node to node in a circular fashion. Only the node that currently possesses the token is permitted to access the network. If the node receiving the token does not have data to transmit, it simply passes the

© 2003 Hewlett-Packard Development Company, L.P.

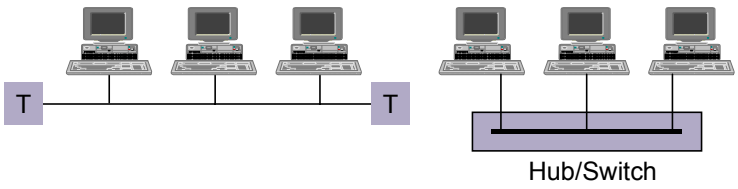
token along to the next node. Token passing provides guaranteed access to every node on the network and is efficient under heavy traffic loads. FDDI and Token Ring networks both use the token passing access method to manage network access.

© 2003 Hewlett-Packard Development Company, L.P.

2–6. SLIDE: Ethernet 802.3 Interface Cards

Ethernet 802.3 Interface Cards

	10Base2	10BaseF	10BaseT	100BaseTX	100BaseFX	1000BaseT	1000BaseSX
Data Rate	10Mbps	10Mbps	10Mbps	100Mbps	100Mbps	1000Mbps	1000Mbps
Log. Topology	Bus	Bus	Bus	Bus	Bus	Bus	Bus
Phys. Topology	Bus	Star	Star	Star	Star	Star	Star
Access	CSMA/CD	CSMA/CD	CSMA/CD	CSMA/CD	CSMA/CD	CSMA/CD	CSMA/CD
Cable Type	Coax	Fiber	Cat 3/5	Cat 5	Fiber	Cat 5	Fiber
Max. Segment	185m	1000m+	100m	100m	412m+	100m	220m+



Student Notes

HP supports a variety of Network Interface Card (NIC) types for the HP 9000 server and workstation families. The next few slides present an overview of the most common NIC card types found in HP boxes today. Each of the standards described here define:

- What cable types are supported
- What cable segment lengths are supported
- That maximum data transmission rate is supported
- What topologies are supported
- What LAN access method is used
- How collisions are handled
- And much more

Ethernet Standards

The network standards shown on the slide above are all variations on the Ethernet/IEEE 802.3 LAN standard. The first Ethernet network was developed at the Xerox PARC research lab in the early 1970s. This was among the first networks ever to use the CSMA/CD access method. In 1980, DEC, Intel, and Xerox banded together to publish what became known as the "DIX Ethernet Standard," which was followed by the official IEEE (Institute of Electrical and Electronic Engineers) 802.3 Standard in 1985; both standards were based on the

© 2003 Hewlett-Packard Development Company, L.P.

CSMA/CD research done at PARC. In the years since 1985, Ethernet has become the most widely used LAN technology.

The original Ethernet IEEE 802.3 standard was based on ThickLAN, or 10base5 coaxial cable, and offered a 10 Mbps transmission speed. Since then, as networking technology has progressed, IEEE has supplemented the original 802.3 standard. The table on the slide lists the most common Ethernet interface card types that HP supports today.

Note that although the various Ethernet specifications support different cable types, transmission speeds, segment lengths, and physical topologies, they all share several features in common. All support the traditional Ethernet frame structure, the CSMA/CD access method, and a logical bus topology.

10Base5	10 Mbps Ethernet specification using thicknet coaxial cable, with a 500-meter maximum segment length. HP stopped supporting 10Base5 for HP 9000s in 1998.
10Base2	10 Mbps Ethernet specification using thinnet coaxial cable, with a 185-meter maximum segment length. 10Base2 networks typically use a physical bus topology. Since twisted-pair has become the preferred cable type in most shops, few interface cards today include a built-in 10Base2 port. Instead, you must attach a 10Base2 LAN "transceiver" to the 15-pin AUI (Attachment Unit Interface) port on the back of the interface card. Then attach a BNC T-connector to the transceiver, which then connects to the thinnet cable run. Be sure to install a thinnet "terminator" on any unused T-connector ports.
10BaseF	10 Mbps Ethernet specification using fiber-optic cable with a maximum segment length of 1000 meters or more depending on the type of cable and transceiver used. "10BaseF" is often used interchangeably with the terms "FOIRL" (Fiber-optic Inter-Repeater Link) and "10BaseFL" (Fiber Link). 10BaseFL is physically cabled in a star topology with pairs of fiber-optic cables radiating out from a central 10BaseFL fiber-optic repeater hub. The fiber-optic cables use two ST (Straight Tip) connectors to attach to a 10BaseFL LAN transceiver, which then attaches to the AUI port on the back of your Ethernet interface card.
10BaseT	10 Mbps Ethernet specification using Cat 3 or 5 twisted-pair cable with a 100-meter maximum segment length. 10BaseT is physically cabled in a star topology with cable radiating out from a central switch or hub. Twisted-pair cable may be attached directly to an RJ45 port on the back of your interface card or to a 10BaseT transceiver on the LAN interface card.
100BaseTX	100 Mbps Ethernet specification using Cat 5 twisted-pair cable with a 100-meter maximum segment length. "100BaseTX" is oftentimes used interchangeably with the abbreviation "100BaseT." 100BaseTX is physically cabled in a star topology, with Cat 5 twisted-pair cable radiating out from a central 100BaseTX hub or switch. The cables attach directly to an RJ45 port on the back of your LAN interface card.
100BaseFX	100 Mbps Ethernet specification using fiber-optic cable with a maximum segment length of 412 meters or more, depending on the type of cable and transceiver (Consult your card's documentation for details). 100BaseFX is

© 2003 Hewlett-Packard Development Company, L.P.

physically cabled in a star topology with fiber-optic cable radiating out from a central 100BaseFX fiber-optic hub or switch. The cables attach directly to the LAN interface card via a Subscriber Connector (SC) duplex connector.

1000BaseT — 1000 Mbps Ethernet specification using Cat 5 twisted-pair cable with a maximum segment length of 100 meters. "1000BaseT" is oftentimes used interchangeably with the term "Gigabit Ethernet." 1000BaseT is physically cabled in a star topology with Cat 5 twisted-pair radiating out from a central switch. Each cable attaches directly to a server's or workstation's LAN card via an RJ45 jack.

1000BaseSX — 1000 Mbps Ethernet specification using fiber-optic cable with a maximum segment length of 220 meters or more, depending on the type of cable and transceiver. 1000BaseSX is physically cabled in a star topology with fiber-optic cable radiating out from a central 1000BaseSX fiber-optic switch. The cables attach directly to the LAN interface card via an SC duplex connector.

NOTE: When you purchase a new interface card, make sure that the card type you buy matches the type of network to which you plan to connect your server or workstation!

Software Requirements

In order to use any of the interface card types listed above, you must install HP's LAN/9000 Link product. You may verify that this product is installed on your system with the **`swlist`** command:

```
# swlist LAN*
```

For the 100 Mbps and 1000 Mbps interfaces listed on the slide, other software bundles are required as well.

NOTE: For the latest list of interface card types supported on your HP 9000, consult HP's web site: **`http://www.hp.com`**. For detailed instructions on installing all types of LAN interface cards, follow the "Networking & Communications" link on the **`http://docs.hp.com`** website.

IEEE 802.3 versus Ethernet

There are some minor differences between IEEE 802.3 and Ethernet LANs. Because both types utilize the same cable media, Ethernet nodes may coexist on the same LAN segment with the IEEE 802.3 nodes. The most significant differences are in the frame format and the electrical grounding of the hardware. All HP 9000 LAN interfaces are able to transmit and receive both IEEE 802.3 and Ethernet frames. The "IP Multiplexing" slide in the next chapter describes how to specify the frame type you wish to use on your network.

© 2003 Hewlett-Packard Development Company, L.P.

Full-Duplex versus Half Duplex

In networks designed according to the original 10Base5 802.3 standard, all hosts on the network connected to a single thicknet cable. The CSMA/CD protocol determined when each host could transmit data on the shared wire. Since all data traveled along one cable, it was impossible for a host to transmit and receive at the same time. This is known as "Half-Duplex Mode" operation.

The advent of twisted-pair cable and Ethernet switches, however, made it possible to offer "Full-Duplex" functionality in an Ethernet environment. Hosts could transmit data over two of the eight wires in a twisted-pair cable, while *simultaneously* receiving data over two of the remaining six wires. Thus, full-duplex mode operation essentially doubles the available bandwidth. Consider 100BaseTX as an example. When operating in half-duplex mode, a 100BaseTX interface card operates at up to 100 Mbps; when operating in full-duplex mode, the very same card may operate at up to 200 Mbps!

In order to be included in the 802.3 standard, a cabling scheme must include some provision for half-duplex, bus-based, CSMA/CD operation. All of the 802.3 standards on the slide except 10Base5 and 10Base2 allow full-duplex operation in addition to the required half-duplex functionality.

- 100BaseTX interface cards use two wires in the twisted-pair cable to transmit and two to receive when operating in full-duplex mode.
- 1000BaseT cards use four wires to transmit, and four to receive when operating in full-duplex mode.
- 10BaseFL, 100BaseFX, and 1000BaseSX all use two parallel fiber-optic cables when operating in full-duplex mode.

In order for full-duplex mode to work properly, both your interface card and the switch to which your host connects must support full-duplex operation!

Auto Negotiation

In order to simplify connectivity between older 10BaseT devices and newer interface cards, all HP 100BaseTX interface cards can operate at either 10 Mbps or 100 Mbps. 1000BaseT interface cards can operate at 10 Mbps, 100 Mbps, or 1000 Mbps. Both card types are capable of operating in either half- or full-duplex mode.

If you wish, you can allow your interface card to "Auto Negotiate" with the switch to which you are attached in order to determine a mutually acceptable speed and duplex setting. If your switch does not support auto-negotiation, HP-UX will automatically sense the link speed and adjust accordingly. It will default to half-duplex operation — even if your switch supports full-duplex functionality!

You can ensure that your link is always configured properly by explicitly setting the card's speed and duplex settings via the **`lanadmin`** command. This procedure will be discussed in detail in the next chapter.

Auto Port Aggregation

The table on the slide shows that 1000BaseT Ethernet interface cards offer 1000 Mbps transmission rates. What can be done if your server needs to move more than 1000 Mbps?

© 2003 Hewlett-Packard Development Company, L.P.

One solution currently available to HP customers is "Auto Port Aggregation." APA is a purchasable software product for HP-UX 11.x, which makes it possible to aggregate multiple interface cards together to form a single, logical, high-bandwidth channel with a single IP address. This offers two major advantages:

- Redundancy. If a link should fail within the APA group, APA provides automatic fail-over for the lost link by redistributing traffic loads across the remaining links within the channel.

- Bandwidth. Using four full-duplex 100BaseTX interface cards in an APA configuration yields an aggregate bandwidth of up to 800 Mbps. Using four 1000BaseSX interface cards in an APA configuration yields an aggregate bandwidth of up to 8Gbps.

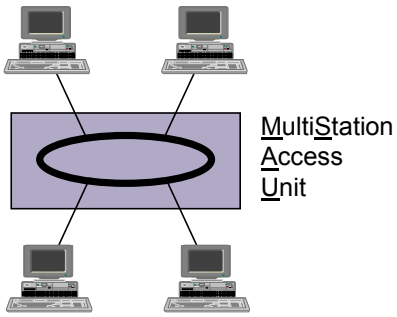
HP has several documents describing Auto Port Aggregation in the Networking and Communications section of the **`http://docs.hp.com`** website.

© 2003 Hewlett-Packard Development Company, L.P.

2–7. SLIDE: Token Ring 802.5 Interface Cards

Token Ring 802.5 Interface Cards

	Token Ring
Data Rate	4 or 16 Mbps
Topology (Logical)	Ring
Topology (Physical)	Star
Access Method	Token
Cable Types	Cat 3/5
Max. Segment	100m



Student Notes

Token Ring 802.5 Standard

Token Ring network technology was originally developed by IBM, but was eventually standardized and endorsed by IEEE in the IEEE 802.5 standard. Today, token ring interface cards are still used primarily in IBM mainframe environments, but may also be found in some HP 9000 boxes that interface with legacy systems.

The following attributes characterize 802.5 networks:

- Bandwidth: 4 Mbps or 16 Mbps
- Logical Topology: Ring
- Physical topology: Star
- Access Method: Token Passing
- Cable Types: IBM Type 1, or Cat 3/5 Twisted-pair
- Maximum Segment Length: 100 meters

The HP Token Ring/9000 product provides a complete link connection to a token ring network. It is fully compliant with IEEE 802.5.

© 2003 Hewlett-Packard Development Company, L.P.

Token Ring networks can be cabled using IBM Type 1 Shielded Twisted-pair (STP) cable with special IBM data connectors, or, more commonly, with standard Cat 3 or 5 Unshielded Twisted-pair (UTP) cabling with RJ45 connectors. HP's Token Ring interface cards provide ports for both cable types, and auto sense which port is currently connected. In either case, the network is connected in a physical star configuration, with cables radiating outward from a central Multi Station Access Unit (MAU or MsAU).

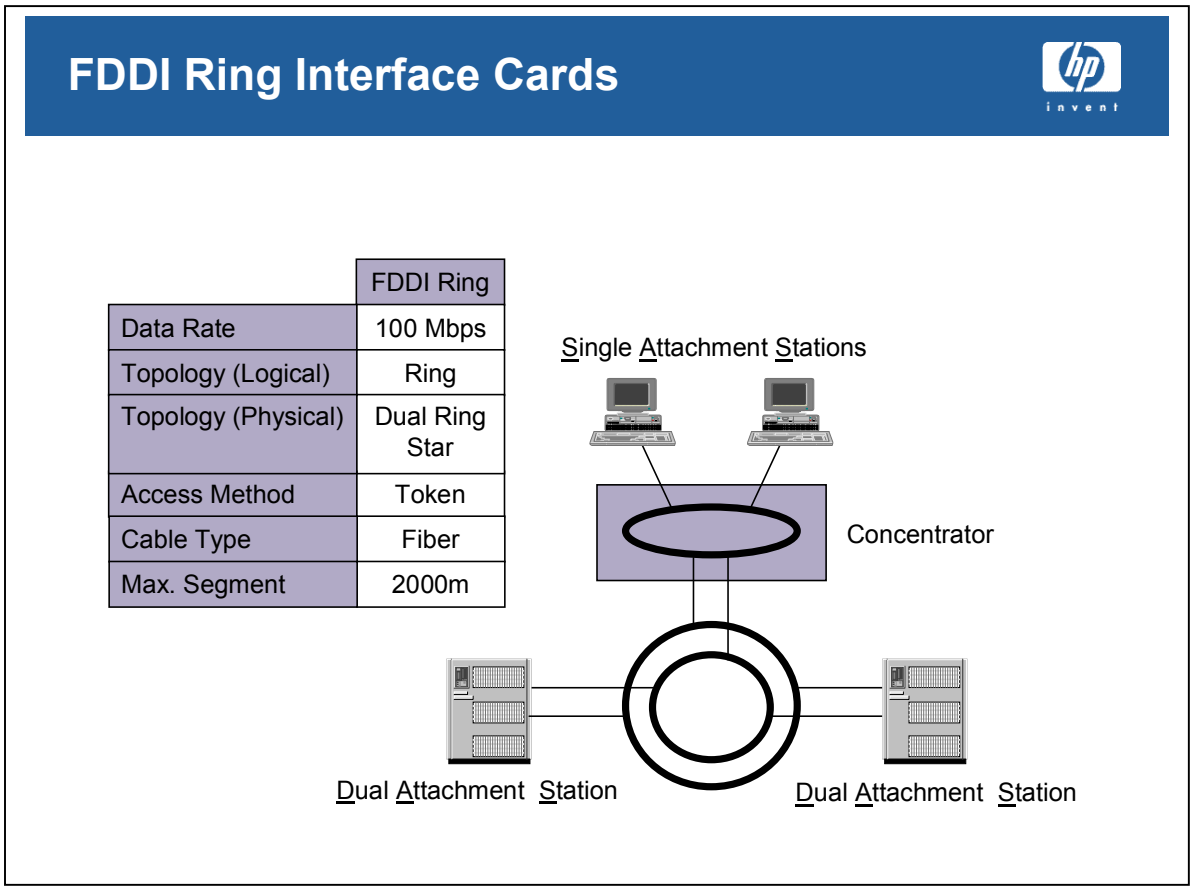
Software Requirements

In order to use a Token Ring interface card on your HP 9000, you must install the Token Ring/9000 software product on your system and include the appropriate driver in your kernel. Check your interface card documentation. Some Token Ring cards require you to configure the ring speed and duplex settings manually; some cards require you to configure these settings via switches on the card itself, while others allow you to make the changes via SAM or the **`lanadmin`** command. See your interface card documentation for details!

NOTE:	For the latest list of interface card types supported on your HP 9000, consult HP's web site: **`http://www.hp.com`**. For detailed instructions on installing all types of LAN interface cards, follow the "Networking & Communications" link on the **`http://docs.hp.com`** website.

© 2003 Hewlett-Packard Development Company, L.P.

2–8. SLIDE: FDDI Ring Interface Cards



Student Notes

The ANSI FDDI standard was developed back in 1986 to provide 100 Mbps, reliable network technology using fiber-optic cable. Even with the advent of fast Ethernet over twisted-pair and fiber, FDDI remains a popular choice for network backbones.

FDDI networks are characterized by the following attributes:

- Bandwidth: 100 Mbps
- Logical Topology: Dual Ring
- Physical topology: Dual Ring, or Star
- Access Method: Token Passing
- Cable Types: Fiber-optic
- Maximum Segment Length: 2000 meters

The FDDI network consists of two independent 100 Mbps rings: the primary and the secondary. The dual-ring approach provides redundancy and the ability to reconfigure the network under fault conditions.

HP supports two different types of FDDI interface cards. Dual-attach (Class A) FDDI interface cards connect to both rings. Single-attach (Class B) FDDI cards attach to a hub-like

© 2003 Hewlett-Packard Development Company, L.P.

FDDI concentrator, which then attaches to both FDDI rings. The concentrator maintains the fault tolerant capability if one ring becomes unusable.

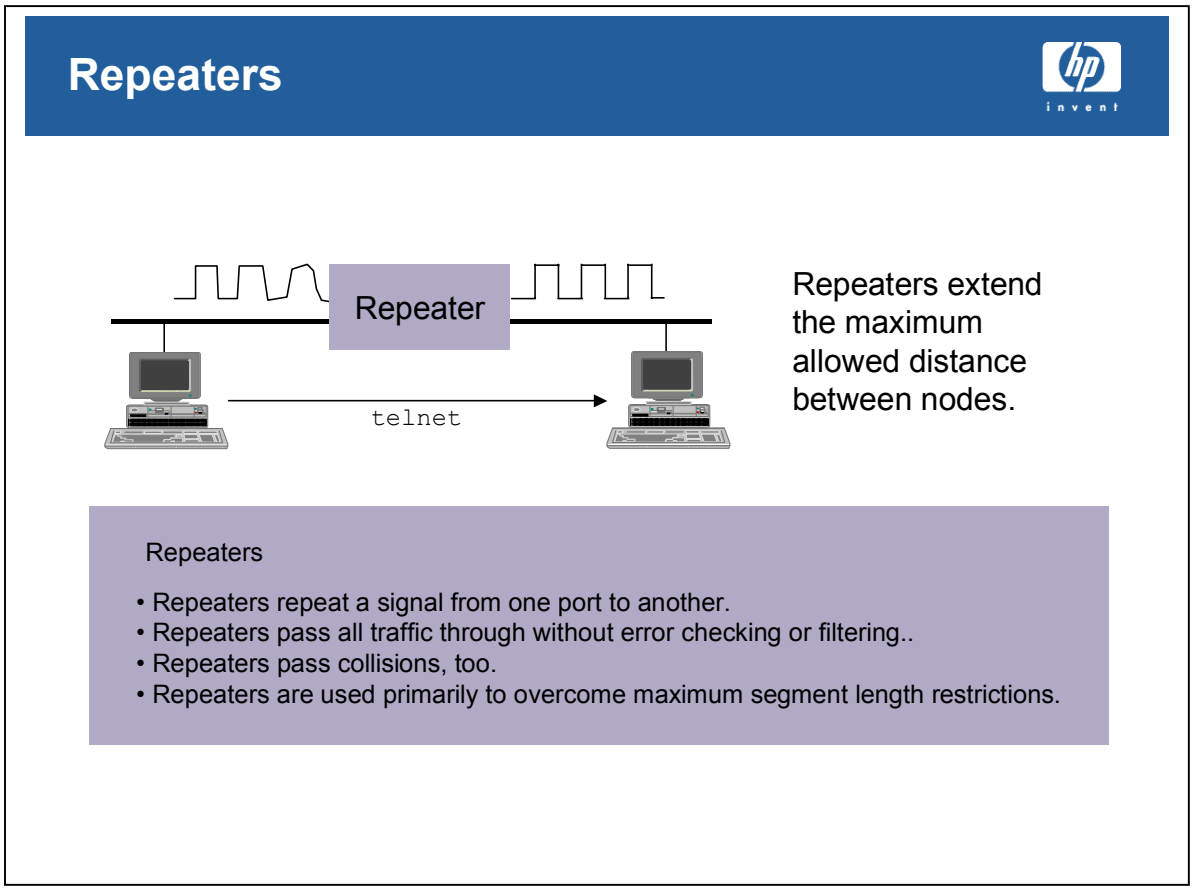
Software Requirements

After physically installing an FDDI card on your system, you must install the FDDI/9000 software product to support it.

NOTE:	For the latest list of interface card types supported on your HP 9000, consult HP's web site: **`http://www.hp.com`**. For detailed instructions on installing FDDI interface cards, follow the "Networking & Communications" link on the **`http://docs.hp.com`** website.

© 2003 Hewlett-Packard Development Company, L.P.

2–9. SLIDE: Repeaters



Student Notes

As an electrical signal travels further and further from the signal source, the signal strength is gradually degraded, which may lead to data corruption. Repeaters provide a mechanism for boosting signal strength and extending the maximum distance between nodes on a network.

Consider the following example: the maximum distance allowed between any two nodes on an Ethernet thinnet segment is 185 meters. A repeater makes it possible to connect two 185m segments to create a single, larger, physical network. The repeater automatically propagates signals from one segment to the other, and vice versa.

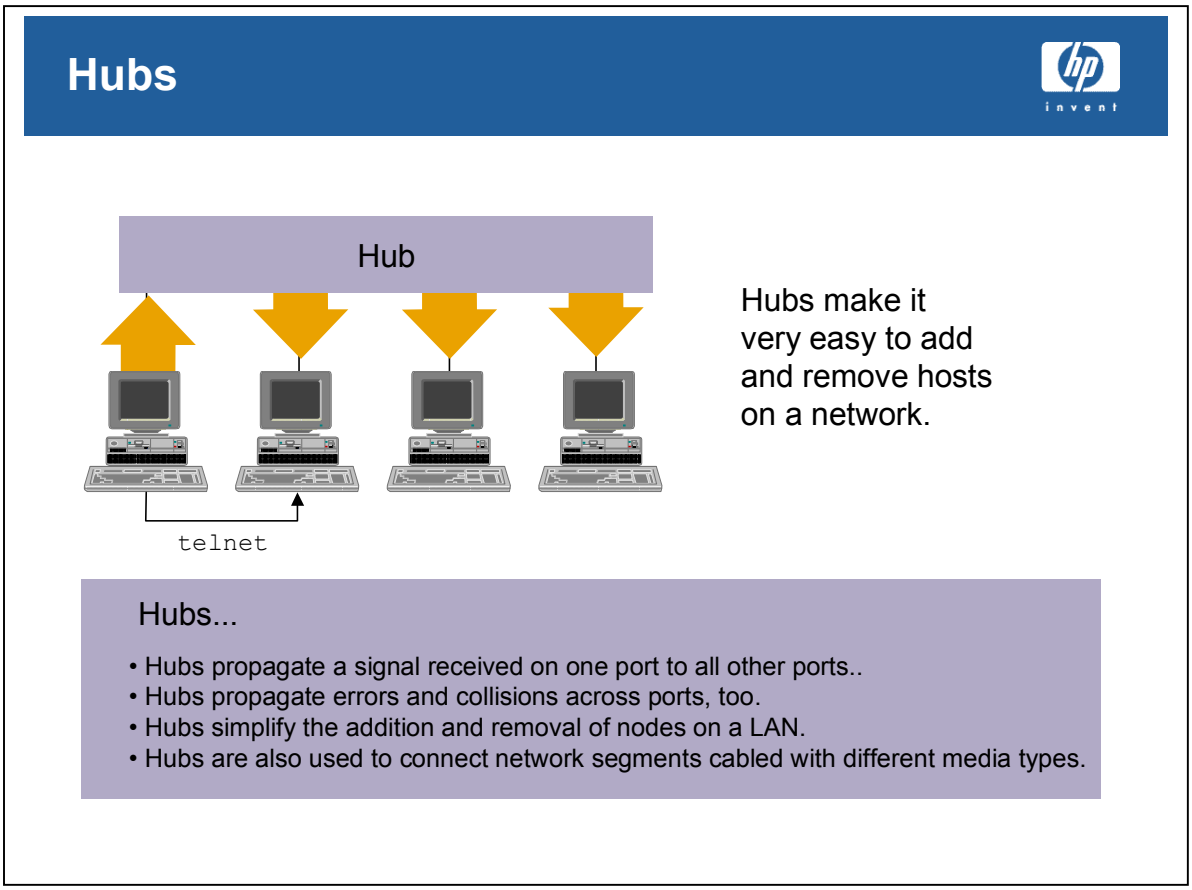
Note that repeaters do nothing to mitigate collisions or errors; they simply propagate signals from port to port.

Question

At which layer of the OSI model does a repeater function?

© 2003 Hewlett-Packard Development Company, L.P.

2–10. SLIDE: Hubs



Student Notes

A hub is simply a multi-port repeater that provides a central connection point for nodes on a network. When a signal is received on one hub port, the hub immediately propagates that signal to the other hub ports. Like repeaters, hubs do nothing to manage collisions. However, they do offer two very important benefits:

- Hosts can be added and removed without disrupting service to other hosts. To add a host, simply run a cable from an available port to the new node. Nodes can also be disconnected from the hub without affecting other hosts on the segment.

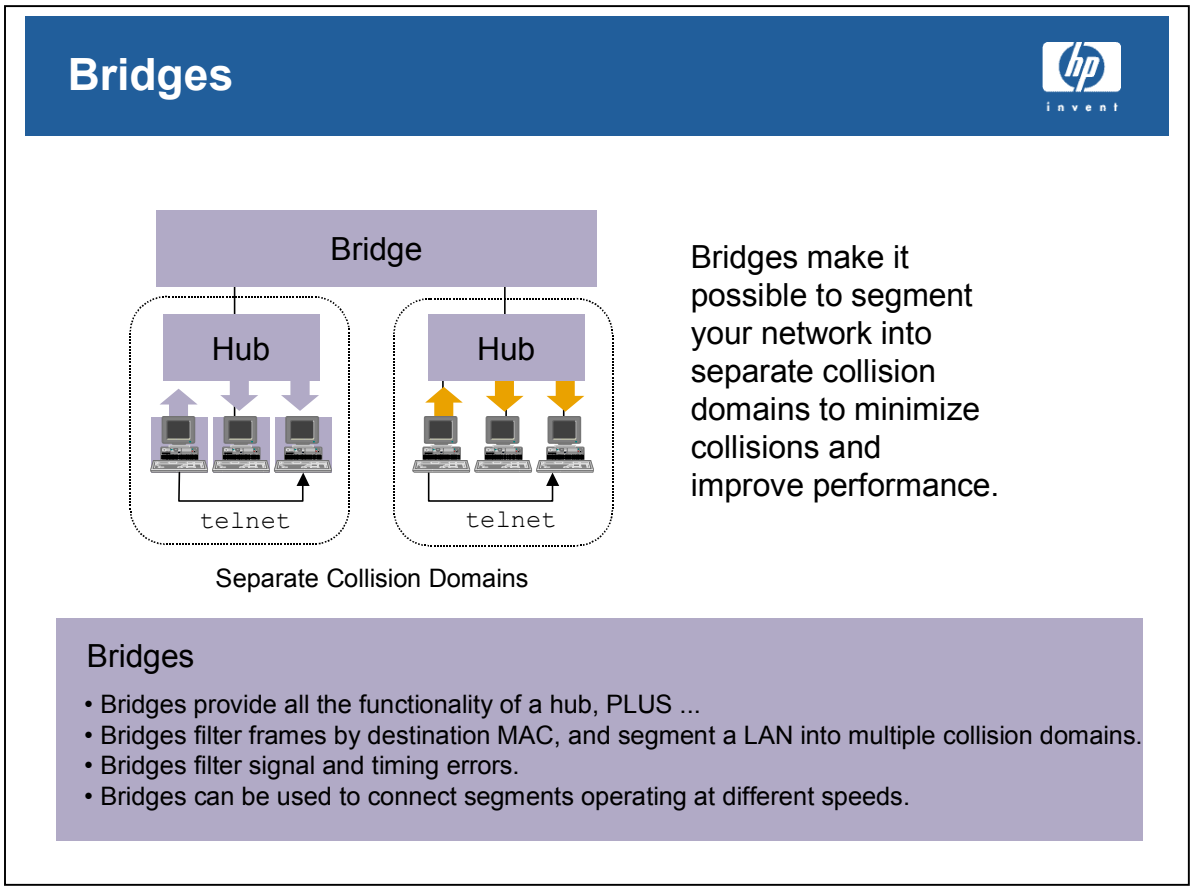
- Hubs are also used to connect hosts cabled using different media types. For instance, a hub may have several thinnet cable ports and several twisted-pair ports. Signals arriving on the twisted-pair ports are automatically propagated to the thinnet ports and vice versa.

Question

At which layer of the OSI model does a hub function?

© 2003 Hewlett-Packard Development Company, L.P.

2–11. SLIDE: Bridges



Student Notes

Bridges, like hubs, can be used to simplify the addition and removal of nodes and pass data between segments that have been cabled using different media types. However, bridges offer several advantages over repeaters and hubs:

- Bridges filter frames by destination MAC and segment a LAN into multiple collision domains.

  On an Ethernet network connected exclusively with hubs and repeaters, no two hosts can transmit simultaneously without causing a collision. All the hosts on the network are members of a single "collision domain." As the number of hosts in a collision domain increases, collisions will likely increase, and performance will be degraded.

  Bridges maintain "bridge forwarding tables" that record which MAC addresses are on each network segment. When a bridge receives a frame, it examines the frame's destination MAC and forwards only that frame to the segment that the destination host is on. This filtering mechanism prevents traffic between hosts on one segment from impacting hosts on other segments and effectively separates a network into two or more collision domains.

© 2003 Hewlett-Packard Development Company, L.P.

- Bridges filter signal and timing errors.

  Occasionally, a malfunctioning interface card may transmit improperly formatted frames. Repeaters and hubs propagate these errors across all ports, which can potentially wreak havoc on the entire network. Bridges reformulate frames before propagating them across ports. This prevents signal or frame errors in one collision domain from affecting other collision domains.

- Bridges can be used to connect segments operating at different speeds.

  Many Ethernet networks today include a heterogeneous mix of older hosts with 10 Mbps interface cards and newer servers with 100 Mbps or even 1000 Mbps interface cards. Bridges use a "store and forward" mechanism to pass data between segments operating at different speeds.

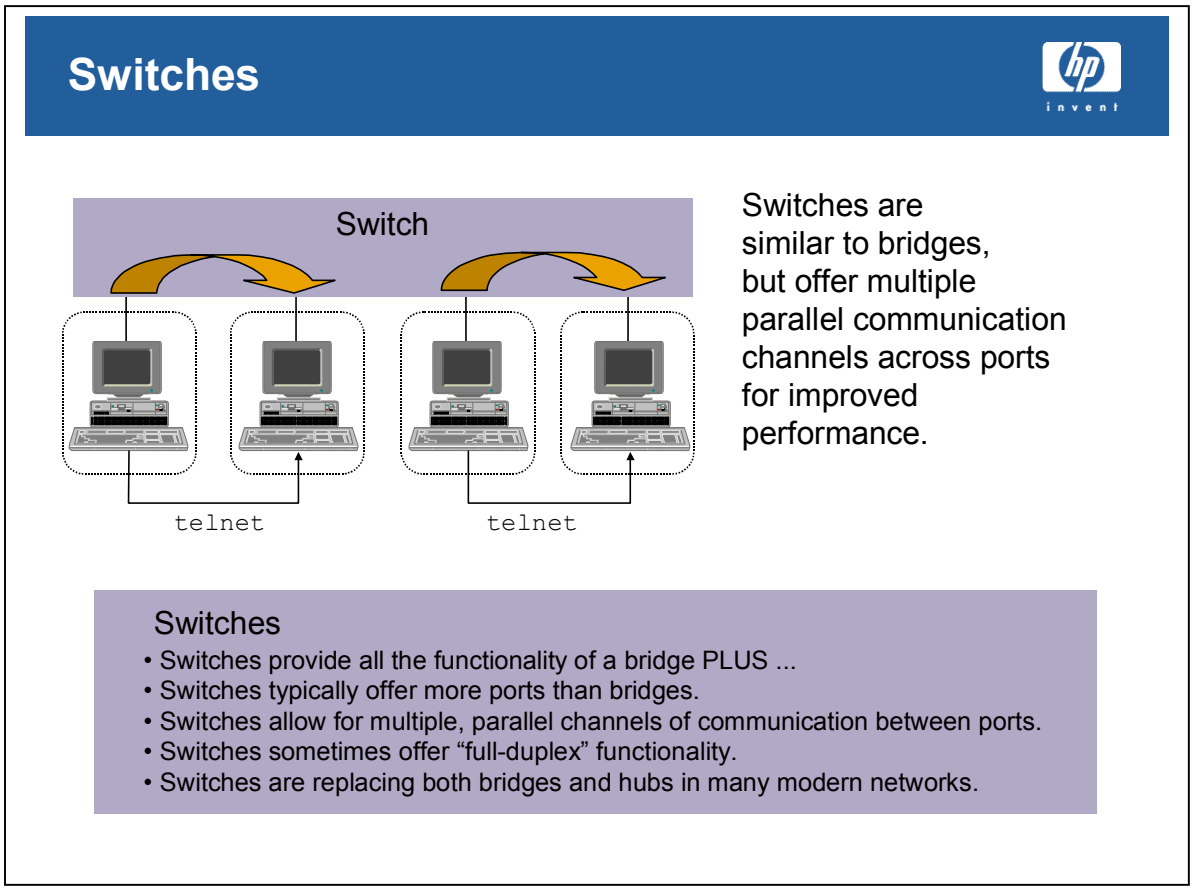
In the past, bridges were typically used to segment departments within a company into separate collision domains to reduce collisions and improve performance. Today, bridges are gradually being replaced by switches, which are described on the next slide.

Question

At which layer of the OSI model does a bridge function?

© 2003 Hewlett-Packard Development Company, L.P.

2–12. SLIDE: Switches



Student Notes

A switch offers many of the same benefits that a bridge offers. Like a bridge, a switch can be used to connect different types of LANs and can filter frames by MAC address in order to divide a busy network into separate collision domains. However, switches offer several important advantages over traditional bridges:

- Switches typically offer more ports than bridges. Traditional bridges only had two ports and were designed to split a network into two separate collision domains. Switches generally offer multiple ports, each of which functions as a separate collision domain.

- Switches allow for multiple, parallel channels of communication between ports. This can dramatically improve performance on many networks.

- Some switches offer “full-duplex” functionality. Host-to-switch connections that are operating in full-duplex mode allow a host to transmit data at the same time that it is receiving data, completely eliminating collisions! This configuration may improve network performance considerably.

- Switches are replacing both bridges and hubs in many modern networks. The price-per-switch-port has dropped in recent years to the point that it is now reasonably economical to provide a dedicated, full-duplex, 100 Mbps switch port for every node on a network.

© 2003 Hewlett-Packard Development Company, L.P.

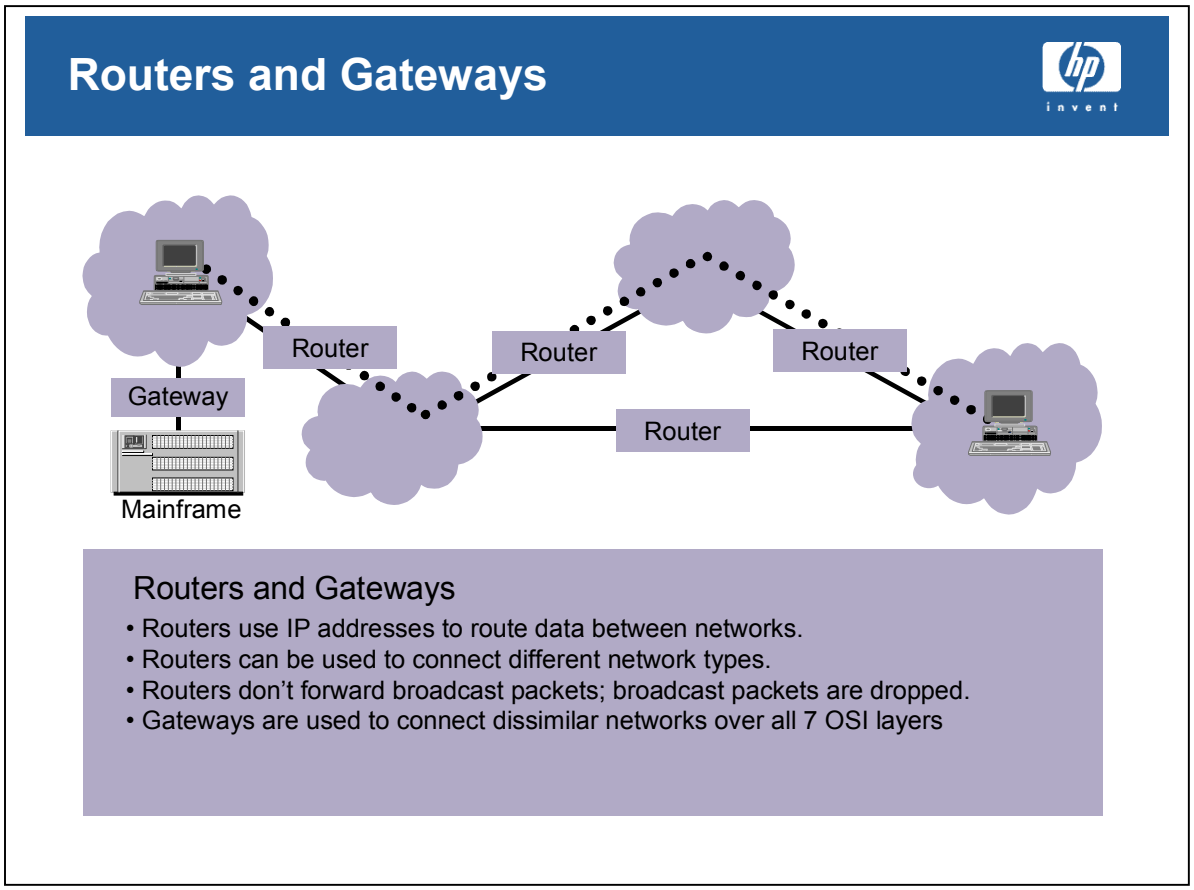
This eliminates collisions and provides a dedicated 100 Mbps link for every workstation and server.

Question

At which layer of the OSI model does a switch function?

© 2003 Hewlett-Packard Development Company, L.P.

2–13. SLIDE: Routers and Gateways



Student Notes

Routers serve the following functions:

- Routers use IP addresses to route data between networks.

  Whereas repeaters, hubs, bridges, and switches are primarily designed to move data within a network, routers are designed to pass data between networks. For instance, in order for a packet of data to travel from a host in your Chicago office to a host in your San Francisco office, the packet must pass through multiple networks. Routers on the Internet determine which route the packet should take to get to the final destination.

  Any HP 9000 system with two LAN cards can serve as a router, but most networks use dedicated rack-mounted routers instead.

- Routers can be used to connect different network types.

  Many organizations today have a heterogeneous network environment. Some departments may be configured as Token Ring networks. Others may be configured as Ethernet networks. Your backbone may be an FDDI network. Your WAN may be an ATM network. Routers typically are used to provide connectivity between different network types.

© 2003 Hewlett-Packard Development Company, L.P.

- Routers do not forward broadcast packets; broadcast packets are dropped.

  Routers provide several mechanisms to improve network performance. Routers treat each port as a separate collision domain, like bridges and switches; however, unlike bridges, routers also filter broadcast traffic. When a broadcast packet arrives on a router port, the router checks the IP network portion of the broadcast address and ensures that the broadcast is propagated only on the desired network. Routers refuse to allow hosts on one network to broadcast traffic to hosts on other networks.

  Some switches these days are also able to filter broadcast traffic.

- Gateways are used to connect dissimilar networks over all 7 OSI layers.

  Gateways are required when you wish to share data across two very different networks that are incompatible at all of the OSI layers. For instance, a gateway would be required in order for HP-UX hosts running TCP/IP over Ethernet to communicate with IBM mainframes on an SNA-based network. An HP 9000 system can operate as an SNA gateway with the SNAplus Link product.

  Since more and more platforms these days use Ethernet and TCP/IP in OSI layers 1 through 3, today's gateways often function in only the top layers of the OSI model. For instance, UNIX hosts use the SMTP protocol over TCP/IP to deliver email, while Microsoft Windows clients use a different email protocol. Since the two platforms use different email protocols, they must communicate with one another through a mail gateway. An HP 9000 system can operate as a UNIX/Microsoft mail gateway using HP's OpenMail product.

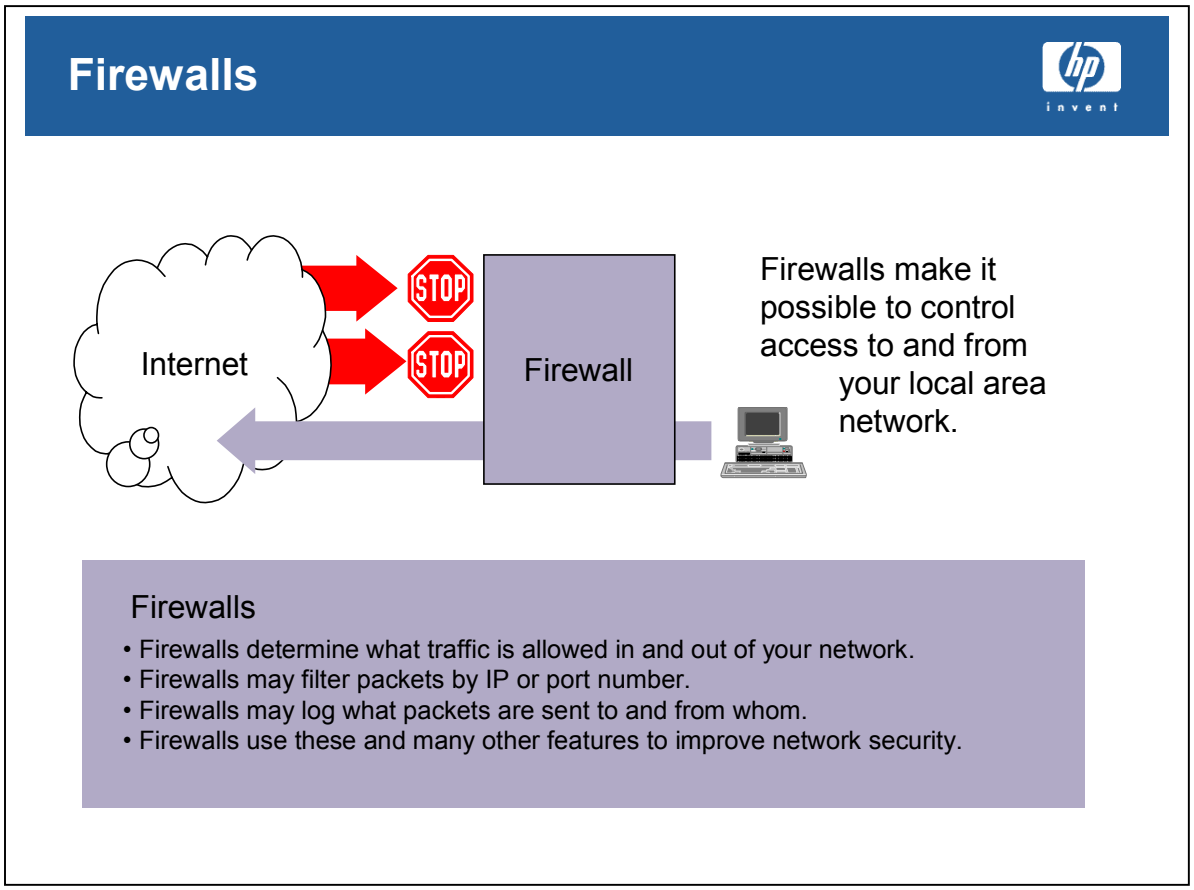
NOTE:	The terms **router** and **gateway** are often used interchangeably. Technically, however, routers operate only at the lower layers of the OSI model, while gateways operate in the upper layers of the OSI model.

Questions

At which layer of the OSI model does a router function?
At which layer of the OSI model does a gateway function?

© 2003 Hewlett-Packard Development Company, L.P.

2–14. SLIDE: Firewalls



Student Notes

Almost every network today includes some sort of firewall to control *who* has access to specific hosts and *when* this access can occur. Most firewalls allow the administrator to filter incoming and outgoing packets based on source and destination IP addresses.

For even more flexibility, most firewalls allow the administrator to control access based on source and destination port numbers. An administrator can choose to allow incoming traffic to reach port number 25 (the port that **`sendmail`** uses to receive incoming email) but can prevent incoming traffic from using **`telnet`** to reach port number 23.

Some firewalls provide even more sophisticated filtering functionality. For example, they look at the contents of incoming email to search for dangerous attachments that might contain viruses.

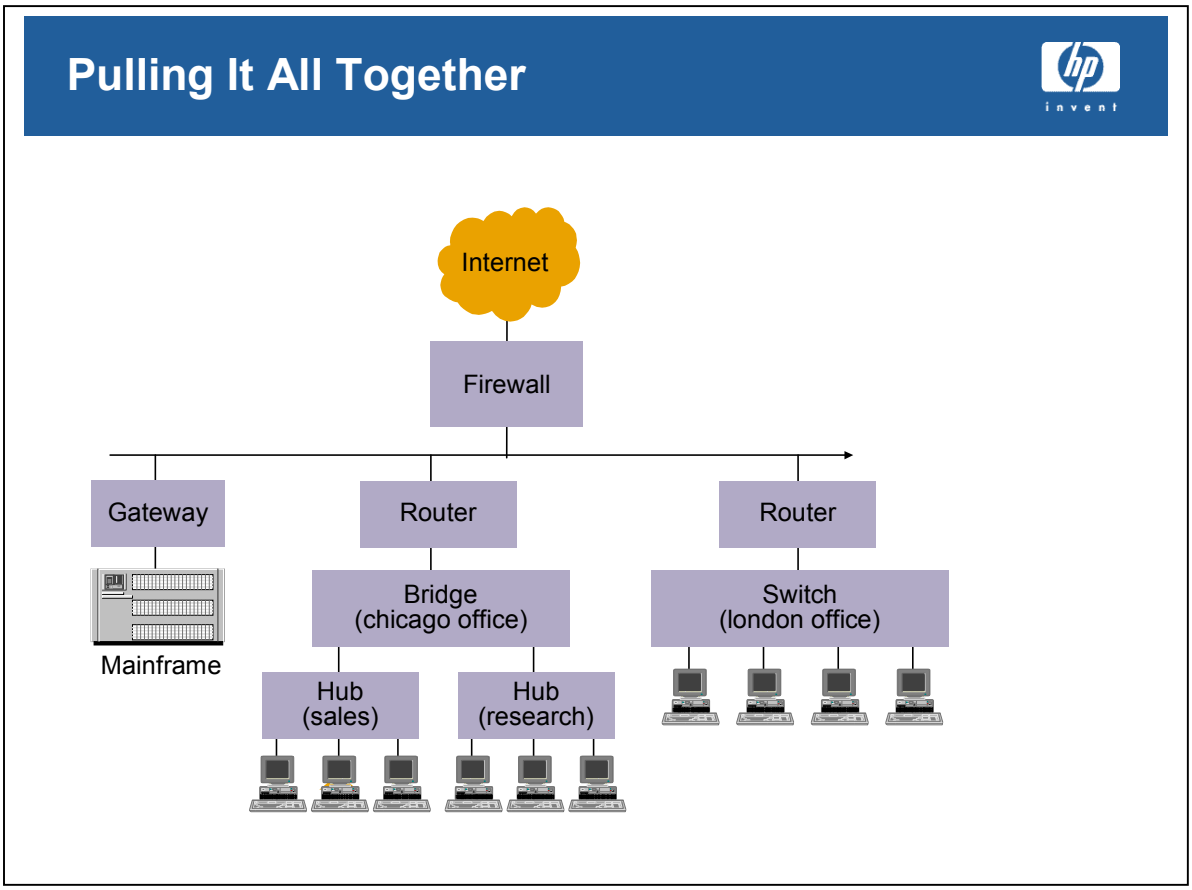
Most firewalls provide some sort of logging mechanism to track which hosts are initiating outbound connections, and which hosts are attempting to get into the internal network.

Question

At which layer of the OSI model does a firewall function?

© 2003 Hewlett-Packard Development Company, L.P.

2–15. SLIDE: Pulling It All Together



Student Notes

The slide shows how hubs, bridges, switches, routers, gateways, and firewalls might be used together in a work environment.

The protocols and devices that were discussed in this chapter are summarized in the following OSI chart:

OSI Layer	Associated Protocols and Devices
7	Gateways, Firewalls
6	
5	
4	
3	Routers
2	IEEE 802.3, IEEE 802.5, FDDI, Bridges, Switches
1	Twisted-pair Cable, Coaxial Cable, Fiber-optic Cable, Repeaters, Hubs

© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 3 — Configuring IP Connectivity

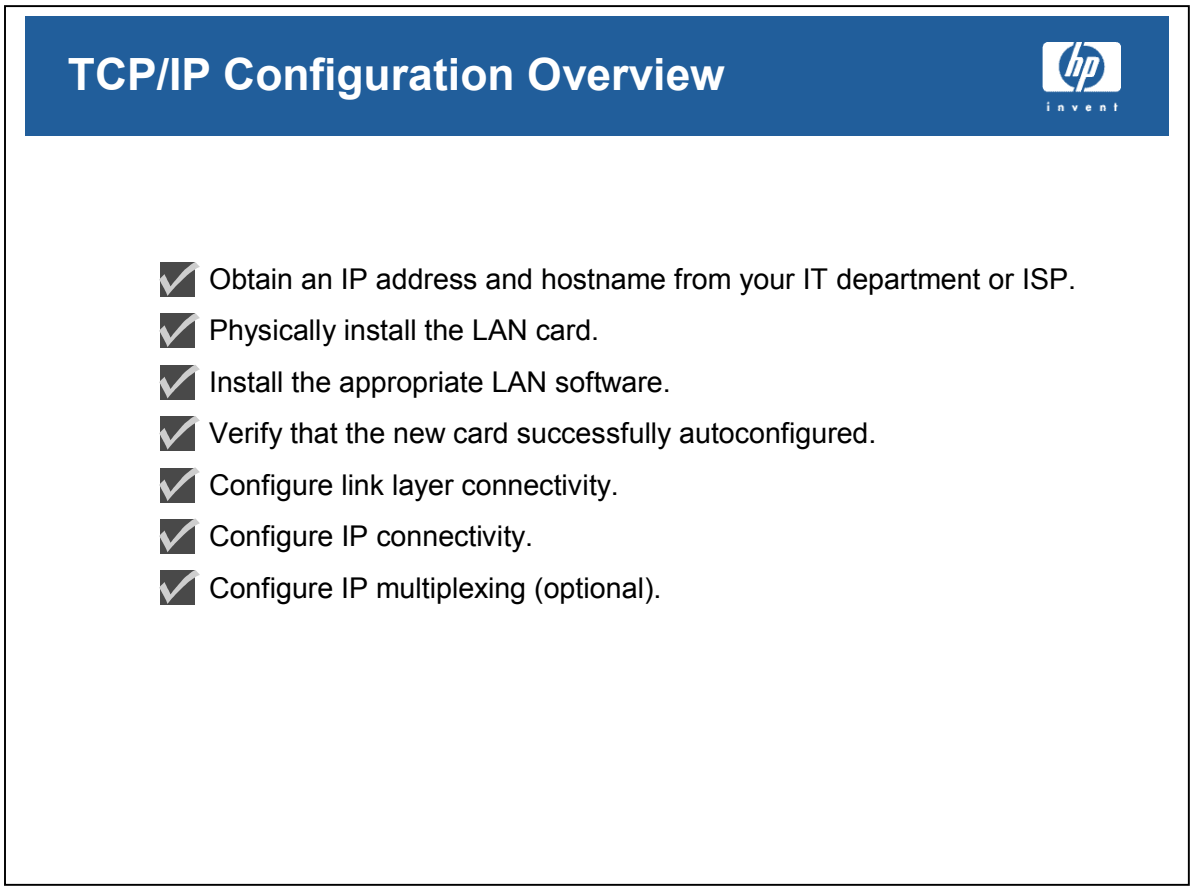
Objectives

Upon completion of this module, you will be able to do the following:

- Configure software and drivers to support a newly installed network interface card.
- Configure link layer connectivity with the **`lanadmin`** command.
- Configure and view the system host name with the **`hostname`** command.
- Configure and view the system IP address and netmask with the **`ifconfig`** command.
- Configure IP multiplexing.
- Configure and use the **`/etc/rc.config.d/netconf`** configuration file.
- Configure the **`/etc/hosts`** configuration file.

© 2003 Hewlett-Packard Development Company, L.P.

3–1. SLIDE: TCP/IP Configuration Overview



Student Notes

Several steps are required to configure an HP-UX host to communicate with a local area network.

First, you must request a valid IP address and host name from your ISP or IT department. Your organization should maintain an up-to-date network map and information table to record which IP addresses and host names have been assigned to which hosts. This minimizes the possibility of duplicate IP addresses, and greatly simplifies network troubleshooting. In your information table, you should record the following information about each host and network device:

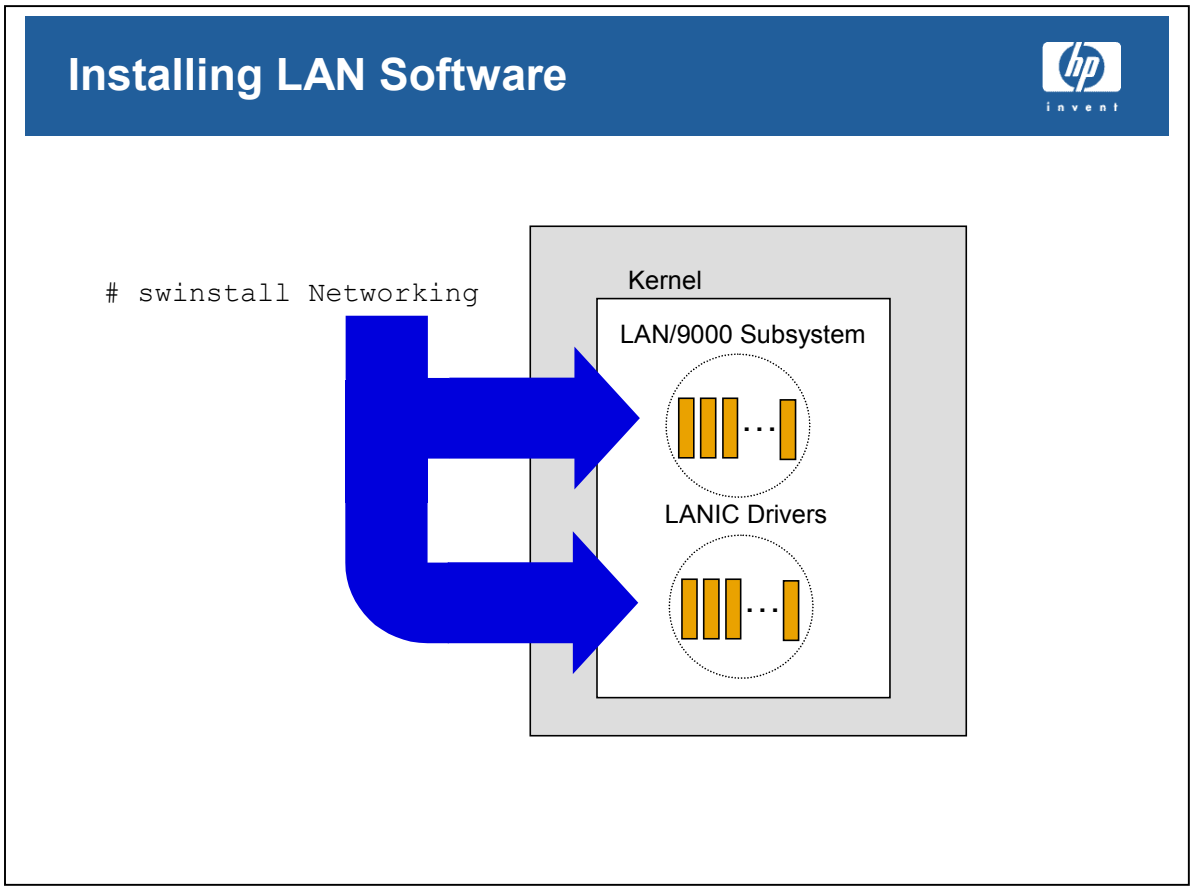
- Manufacturer
- Model number
- OS type and version
- LAN card type
- Host name
- IP Address
- MAC Address
- Administrator name

© 2003 Hewlett-Packard Development Company, L.P.

After obtaining an IP and host name, you are ready to install and configure your interface card! The slide above overviews the required steps, and the remaining slides in the chapter will explain the details.

© 2003 Hewlett-Packard Development Company, L.P.

3–2. SLIDE: Installing LAN Software



Student Notes

The first step in configuring a connection to a local area network is to physically install a LAN interface card. For the latest list of LAN interface cards supported on your HP 9000, check the HP web site at **`http://www.hp.com`**.

Installing the Networking Product

In order to use your new interface card, you will need to install the Networking product on your system. Among other things, the Networking product includes the LAN/9000 and DLPI/9000 kernel subsystems that allow your system to communicate with TCP/IP networks. The Networking product comes standard with HP-UX and was probably included in your original OS install. Use the **`swlist`** command to verify that the Networking product exists on your system:

```
# swlist -l product Networking

  Networking B.11.11 HP-UX_Lanlink_Product
```


© 2003 Hewlett-Packard Development Company, L.P.

If the Networking product is missing, insert the CoreOS CD that came with your system and run the **swinstall** graphical user interface to install the product:

```
# swinstall      (follow the intuitive GUI menus that follow)
```

The Networking product includes all of the software necessary to configure and use a standard Ethernet interface card. If, however, you are using FDDI, Token Ring, 100VG, or other types of LAN cards, it may be necessary to load additional products on your system. Consult your LAN card documentation for more information.

Configuring Kernel Subsystems and Drivers

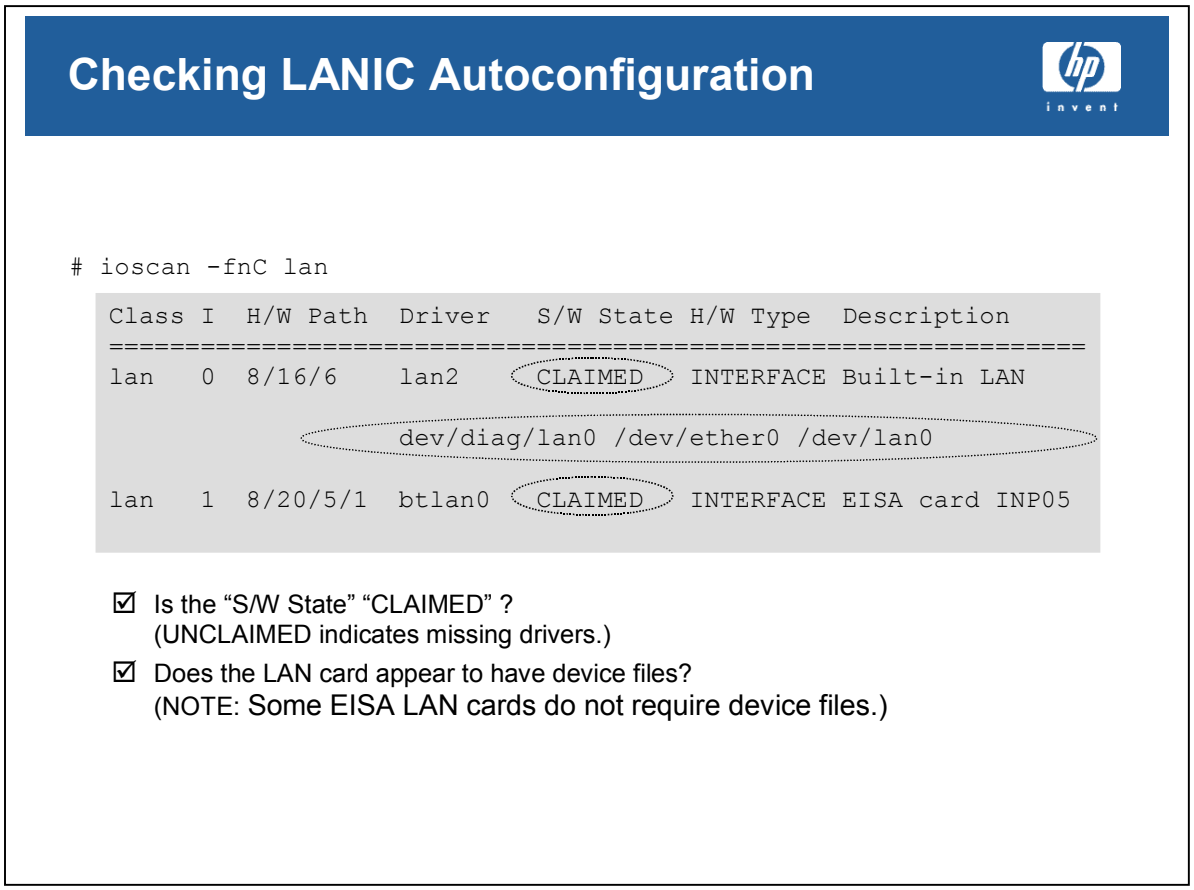
Installing the Networking product *should* automatically configure the LAN/9000 and DLPI/9000 subsystems in the kernel, as well as the drivers required for a 10baseT interface card. However, if **ioscan -fnC lan** shows your LAN card as **UNCLAIMED**, you may need to configure the LAN drivers manually and regenerate your kernel. Consult your documentation to determine which drivers and subsystems are required to support your LAN card.

SAM provides the easiest method for configuring kernel drivers and subsystems:

```
# sam --> Kernel Configuration --> Subsystems
                               --> Drivers
```


© 2003 Hewlett-Packard Development Company, L.P.

3–3. SLIDE: Checking LANIC Autoconfiguration



Student Notes

Assuming the proper drivers are configured in your kernel, HP-UX should automatically recognize new LAN interface cards, and auto-configure hardware paths and device files during the system boot process. You can check the auto-configuration via the **`/usr/sbin/ioscan -funC lan`** command.

Check the **`ioscan`** output for the following:

- Does the card appear at all in the output? If not, the card may not be seated properly in its slot.
- Does the card appear to be **`CLAIMED`**? If not, the card's kernel driver is probably missing. Return to the previous slide to learn how to configure drivers in the kernel.
- Does the card have the necessary device files? Most LAN cards will not function without device files. Assuming the LAN card's driver is configured in the kernel, you can create device files for your LAN card via **`/usr/sbin/insf -eC lan`**. Note that some EISA LAN cards, such as the 100BT LAN card shown on the slide, do not require device files.

© 2003 Hewlett-Packard Development Company, L.P.

Diagnostic Device Files

Diagnostic device files are required by the LAN diagnostic tools **linkloop** and **lanadmin**. These and other troubleshooting tools will be presented later in this course. Check the diagnostic device files with **ll:**

```
# ll /dev/dlpi*
crw-rw-rw-  1 bin    bin     72 0x000077 May 11 15:32 /dev/dlpi
crw-rw-rw-  1 bin    bin    119 0x000000 May 11 15:32 /dev/dlpi0
crw-rw-rw-  1 bin    bin    119 0x000001 May 11 15:32 /dev/dlpi1
crw-rw-rw-  1 bin    bin    119 0x000002 May 11 15:32 /dev/dlpi2
crw-rw-rw-  1 bin    bin    119 0x000003 May 11 15:32 /dev/dlpi3
crw-rw-rw-  1 bin    bin    119 0x000004 May 11 15:32 /dev/dlpi4
```

Recreate the diagnostic device files with **insf**:

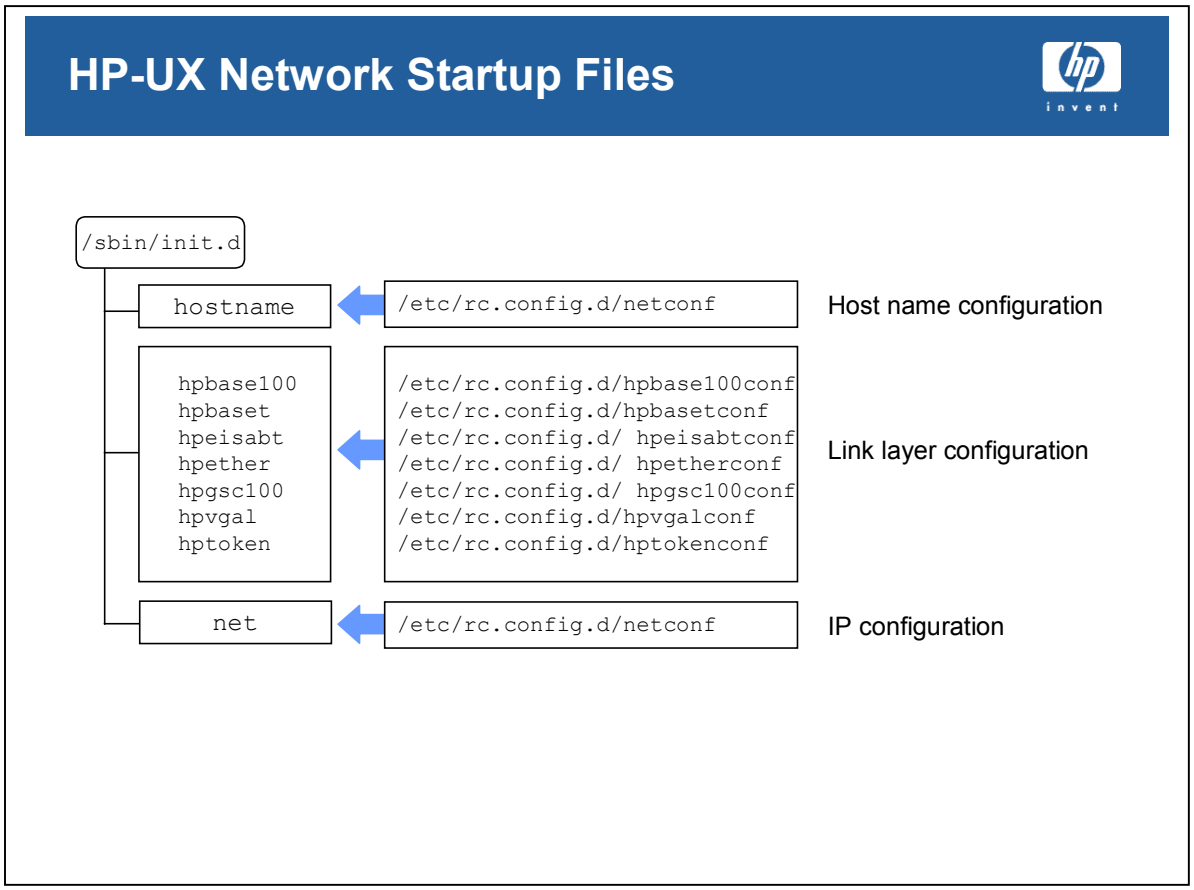
```
# cd /dev
# insf -d dlpi -e
insf: Installing special files for pseudo driver dlpi
```

LAN Interface Cards with Two LAN Connectors

Most of the built-in, and all additional LAN interfaces, have two LAN connectors. Some older machines had an AUI and a ThinLAN port, but all new machines are shipped with an AUI and an EtherTwist port. Older LAN interfaces had a jumper to select one of the two ports. The new LANICs detect the connected port automatically.

© 2003 Hewlett-Packard Development Company, L.P.

3–4. SLIDE: HP-UX Network Startup Files



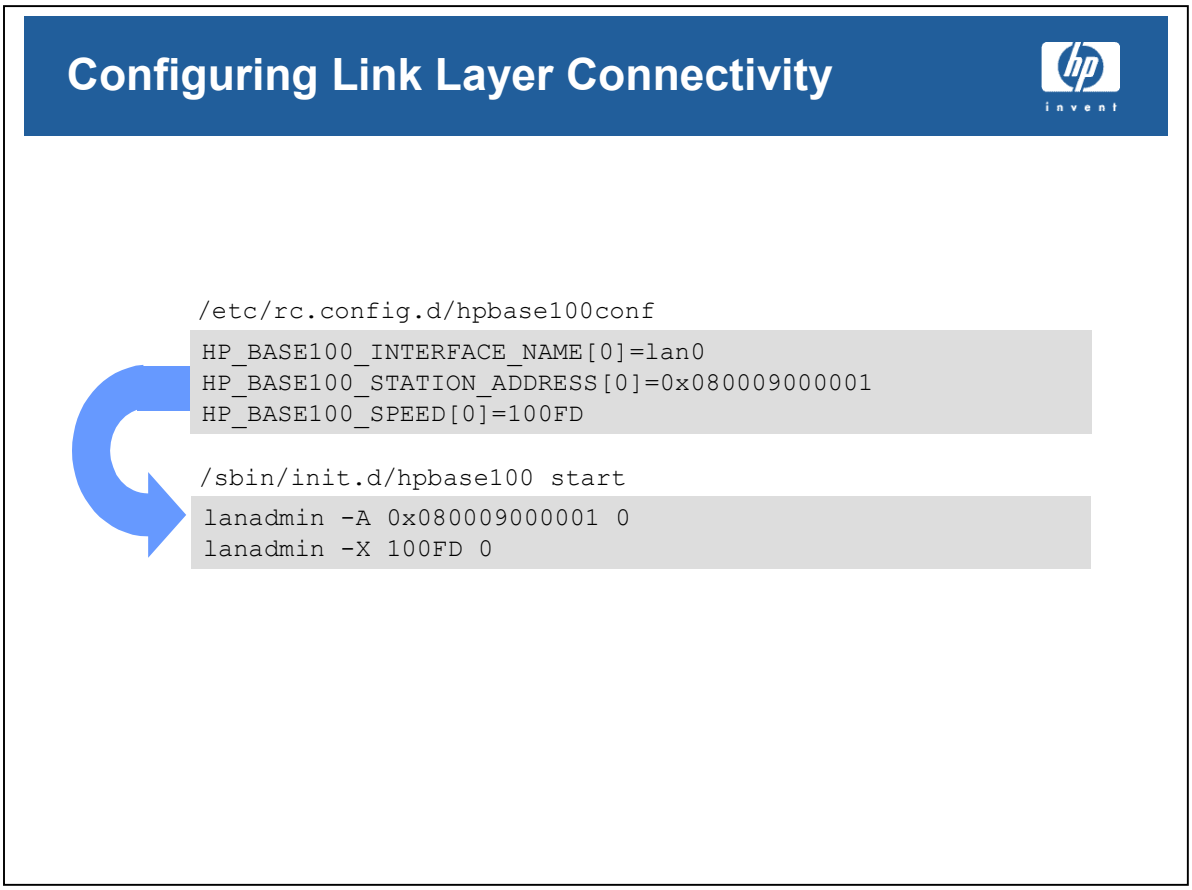
Student Notes

During the system startup process, the **`/sbin/rc`** program executes several scripts in the **`/sbin/init.d`** directory. These **`/sbin/init.d`** scripts read configuration parameters from a collection of configuration files in the **`/etc/rc.config.d`** directory, and initialize your network connection. The remaining slides in this chapter will describe the parameters in each of these configuration files in detail.

WARNING: Never modify the scripts in **`/sbin/init.d`**! Startup script configurable parameters should be modified *only* with the configuration files in **`/etc/rc.config.d`**.

© 2003 Hewlett-Packard Development Company, L.P.

3–5. SLIDE: Configuring Link Layer Connectivity



Student Notes

The **`/sbin/init.d`** directory contains several scripts that initialize data link layer parameters associated with your LAN interface cards. Since different interface cards support different configurable parameters, there are separate scripts for each supported interface card type. The sample script and configuration file shown on the slide are used to configure HP 100BaseT PCI interface cards. Check your documentation to determine which configuration file your LAN card uses.

Configuring the `/etc/rc.config.d/*` Files

The parameters available in the configuration file vary somewhat from interface card to interface card, but some are common across many card types. Note that each of these variable names will be preceded by a string identifying the LAN card type.

`INTERFACE_NAME` Identifies the name of the LAN card defined by the current block of variables (**`lan0`**, **`lan1`**, etc.). Use the **`lanscan`** command to list the recognized LAN interfaces on your system.

`STATION_ADDRESS` Sets the LAN card's MAC address. If left blank (recommended!), the card will use the preset MAC address coded on the interface card by the manufacturer. If you choose to override the preset MAC address,

© 2003 Hewlett-Packard Development Company, L.P.

you must use a 12-digit hexadecimal number, preceded by a "0x" prefix. Use this feature with caution!

DUPLEX Many LAN cards can operate in either "full-duplex" mode, which permits the host to transmit and receive simultaneously, or "half-duplex" mode, which prevents the host from transmitting and receiving simultaneously. Check with your IT department to determine the appropriate setting for your environment and change the DUPLEX value accordingly. Most cards recognize two values: "FULL" or "HALF."

SPEED Some LAN cards may operate at 10 Mbps (if connected to a 10BaseT network), 100 Mbps (if connected to 100BaseT network), or even 1000 Mbps (if connected to a 1000BaseT network). In most cases, the card will "auto-sense" and set the appropriate speed setting automatically. On some cards, however, you may override the default speed via the **SPEED** variable and the **-X** option on **lanadmin**.

Typically, startup scripts that consult the **SPEED** variable do not consult the **DUPLEX** variable. Instead, both parameters are defined via a single variable using one of the following:

```
SPEED[0]=100FD
SPEED[0]=100HD
SPEED[0]=10FD
     SPEED[0]=10HD
     SPEED[0]=auto_on # "autosense"
```

Here again, you should ask your IT department which setting to use in your environment.

If you have multiple interface cards on your system, you may replicate the block of variable definitions in this file, one block for each interface card. Change the index following each variable in the second block of lines to **[1]**s, in the third block of lines to **[2]**s, and so on. Then fill in the variable values as appropriate.

Executing `lanadmin` via the `/sbin/init.d/*` Scripts

When your system boots, it automatically executes the **/sbin/init.d** scripts, which, in turn, read the configuration files in **/etc/rc.config.d**. The **/sbin/init.d** scripts use the **lanadmin** command to set the link layer parameters that you have defined.

The list of parameters that may be configured via **lanadmin** varies from card to card. Consult your documentation for more information. The general syntax for **lanadmin** is consistent. The first option/argument pair determines which parameter you wish to configure, and the last argument identifies the card you wish to configure. At HP-UX 10.20, the card is identified by the "Network Management ID (NMID) Number", while HP-UX 11.x requires you to specify the card to configure by "Physical Point of Attachment (PPA) Number". Both of these values may be obtained via the **lanscan** command. Note that the **/etc/rc.config.d/hpbase100conf** configuration file simply takes the interface name as an argument and automatically determines the PPA/NMID numbers as needed. Consider

© 2003 Hewlett-Packard Development Company, L.P.

the following examples. The first example below shows the procedure required at 11.x, while the second block of lines shows the procedure required at 10.20:

```
# lanscan -ip                    # determine the PPA number (for 11.x)
# lanadmin –A 0x080009000001 0   # set the MAC address for card at PPA 0
# lanadmin –X 100FD 0            # enable 100Mbit, full-duplex

# lanscan -in                    # determine the NMID number (for 10.20)
# lanadmin –A 0x080009000001 1   # set the MAC address for card at NMID 1
# lanadmin –X 100FD 1            # enable 100Mbit, full-duplex
```

lanadmin may also be used to check the currently defined parameters for one of your interface cards. Again, **lanadmin** requires a PPA number at 11.x, or an NMID number at 10.20:

```
# lanadmin –a 0                  # check PPA 0s MAC address
# lanadmin –s 0                  # check PPA 0s speed setting
# lanadmin –x 0                  # check PPA 0s duplex setting
                                    (not supported on all cards)
```

A Note about Non-Ethernet LAN Interface Cards

The discussion on this slide concentrates on Ethernet interface cards, since those are the most common LAN interfaces found on HP systems today. Other interface cards have similar configuration files in **/etc/rc.config.d** that are used to define interface card specific parameters. For instance, installing the Token Ring/9000 software product on your system creates a file called **/etc/rc.config.d/hptokenconf**. This file includes several token ring specific parameters:

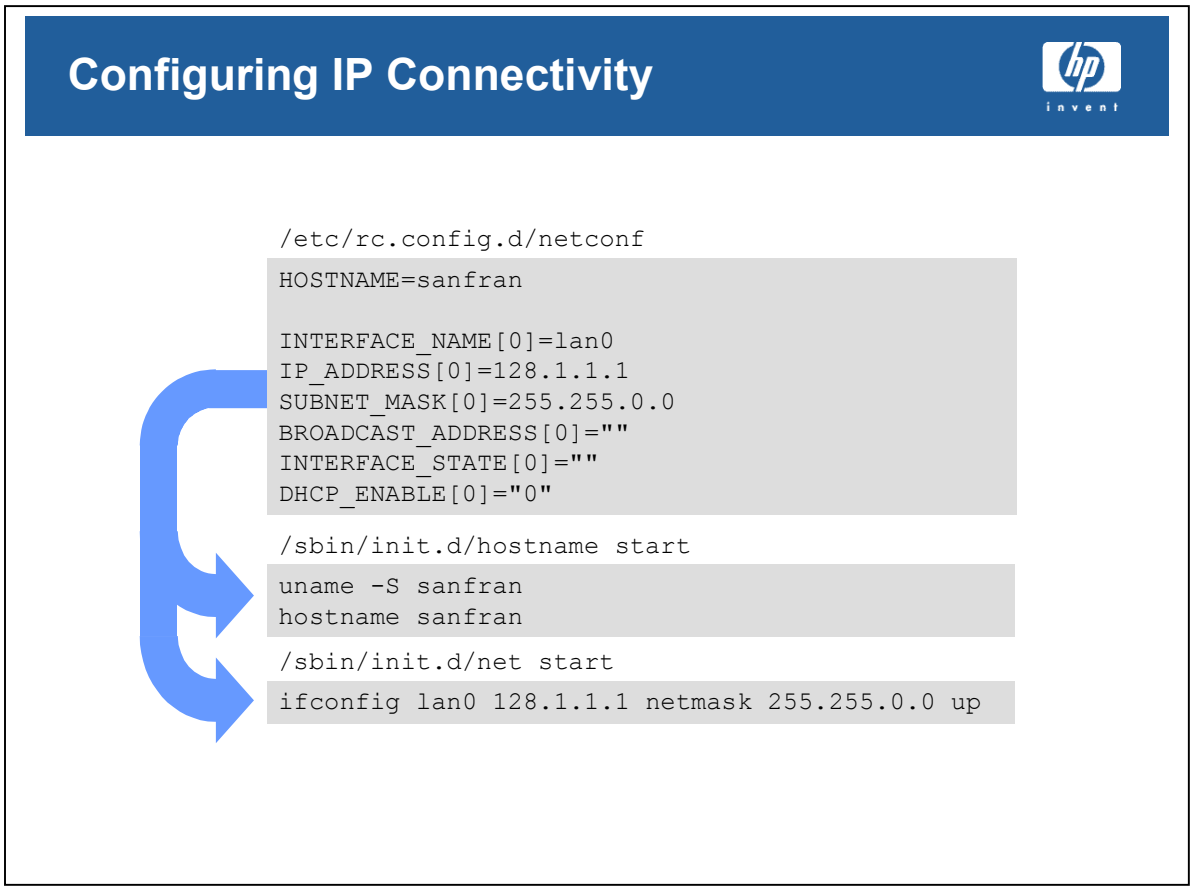
```
HP_TOKEN_INTERFACE_NAME[0]     # which card does this apply to?
HP_TOKEN_STATION_ADDRESS[0]    # MAC address
HP_TOKEN_MTU[0]                # maximum transmission unit
HP_TOKEN_RING_SPEED[0]         # 4 Mbits or 16 Mbits per second?
```

The **/sbin/init.d/hptoken** startup script uses these variable values as arguments to the **lanadmin** command to configure your system's token ring interface cards fully during the system boot process.

Other interface cards use other configuration files with different variable parameters. Consult your documentation for more information.

© 2003 Hewlett-Packard Development Company, L.P.

3–6. SLIDE: Configuring IP Connectivity



Student Notes

`/etc/rc.config.d/netconf` file is the primary TCP/IP configuration file in HP-UX. This file is read by several different startup scripts that configure everything from the system host name to the **`gated`** dynamic routing protocol daemon. For now, we will concentrate on the first half of the file which defines the system host name and IP address.

Modifying `/etc/rc.config.d/netconf`

The first block of lines in the **`netconf`** file defines some general system parameters. Change the **`HOSTNAME`** variable if you wish to change the system host name. The other two parameters, **`OPERATING_SYSTEM`** and **`LOOPBACK_ADDRESS`**, should never be changed.

```
HOSTNAME="sanfran"
OPERATING_SYSTEM=HP-UX
LOOPBACK_ADDRESS=127.0.0.1
```

Further down in the file, look for the following block of lines:

```
INTERFACE_NAME[0]=lan0        # use the name shown in lanscan
IP_ADDRESS[0]=128.1.1.1       # set the IP address here
SUBNET_MASK[0]=255.255.0.0    # netmask in dotted decimal
```


© 2003 Hewlett-Packard Development Company, L.P.

```
BROADCAST_ADDRESS[0]=""       # broadcast address may be defaulted
INTERFACE_STATE[0]=""         # bring card "up" at boot? default=up
DHCP_ENABLE[0]="0"            # if "1", DHCP will set the IP address
```

If you have multiple LAN cards, copy this block of lines and change the variable indices. Then change the variable values as appropriate. Appending the sample block of lines below to the **netconf** file would assign IP address 192.1.1.1 to the lan1 interface card:

```
INTERFACE_NAME[1]=lan1
IP_ADDRESS[1]=192.1.1.1
SUBNET_MASK[1]=255.255.255.0
BROADCAST_ADDRESS[1]=""
DHCP_ENABLE[1]="0"
```

Setting the System Host Name with /sbin/init.d/hostname

When the system boots to run level 1, the **/sbin/init.d/hostname** script sources **/etc/rc.config.d/netconf** and sets the system host name.

Technically, UNIX systems may be identified by two different host names. The "UNIX-to-UNIX copy" (UUCP) service identifies hosts by UUCP host name. The UUCP host name may be both set and verified via the **uname** command:

```
# uname -S sanfran    # set the uucp hostname
# uname -n            # view the uucp hostname
```

Most other network services identify hosts by their internet host names. You may set and view the Internet host name via the **hostname** command:

```
# hostname sanfran    # set the internet hostname
# hostname            # view the system hostname
```

Theoretically the uucp host name may be different from the Internet host name. However, HP strongly recommends that the two host names be identical. The **/sbin/init.d/hostname** startup script guarantees this by using the **HOSTNAME** variable as an argument to both **uname -S** and **hostname** during the system startup process.

Setting IP Addresses with /sbin/init.d/net

When the system reaches run-level 2, the **/sbin/init.d/net** script sources **/etc/rc.config.d/netconf** and sets your system IP address(es) and netmask(s) using the **ifconfig** command. The most common **ifconfig** syntax is shown below:

```
# ifconfig lan0 up                                  # allow traffic on lan0
# ifconfig lan0 down                                # deny all traffic on lan0
# ifconfig lan0 128.1.1.1 netmask 255.255.0.0 up    # change lan0's IP/netmask
```

If you specify **ifconfig *interface*** with no other parameters, **ifconfig** displays the name of the enabled network interface, the IP address, subnet mask, broadcast address, and other flags.

© 2003 Hewlett-Packard Development Company, L.P.

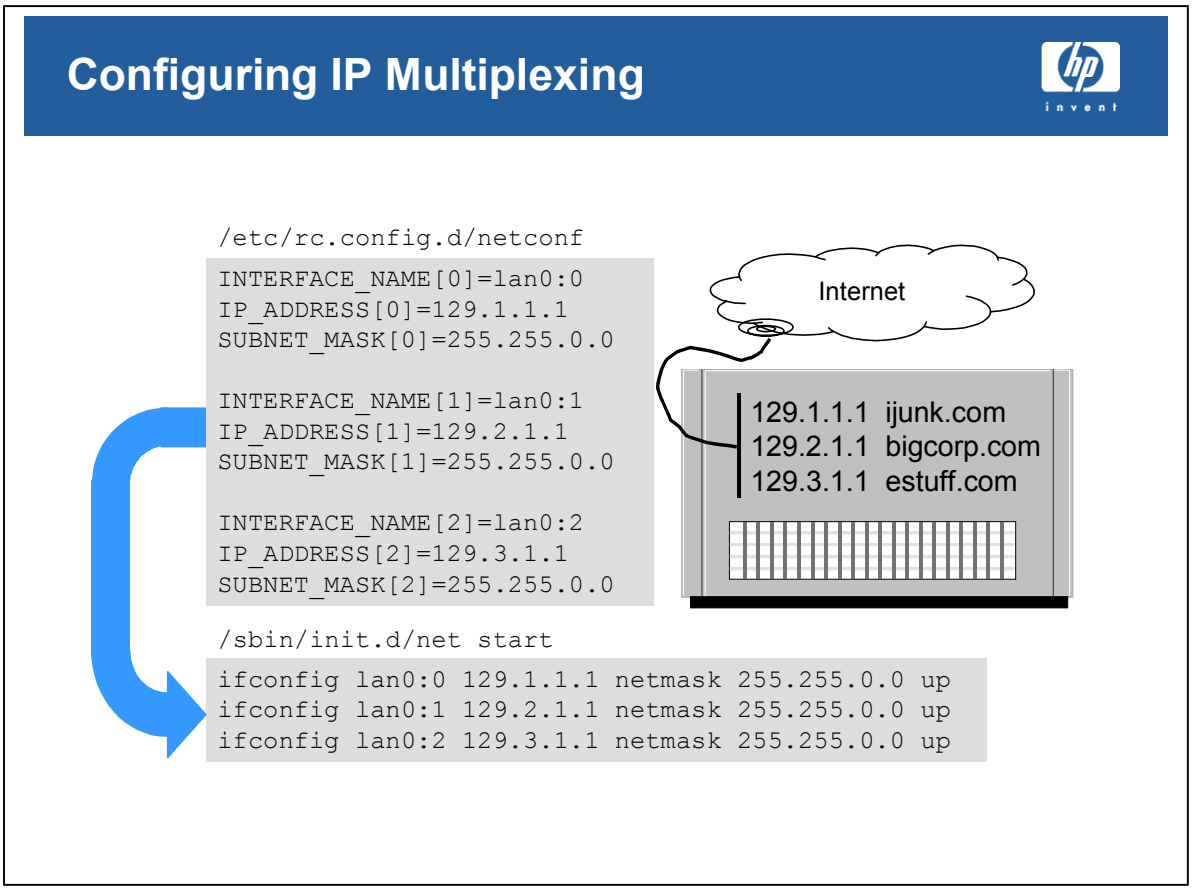
```
# ifconfig lan0
   lan0: flags=863<UP,BROADCAST,RUNNING,MULTICAST>
         inet 128.1.1.1 netmask ffff0000 broadcast 128.1.255.255
```

Watch particularly for the **UP** flag in the **ifconfig** output. If **ifconfig** doesn't explicitly state that a card is **UP**, the card will neither send nor receive any IP traffic!

CAUTION: Many applications (including CDE!) are dependent on the IP address and the host name. Ideally, you should shut down all applications before changing your IP address or host name. Perhaps the simplest approach is to make the desired changes in **/etc/rc.config.d/netconf**, then reboot to restart all of your applications.

© 2003 Hewlett-Packard Development Company, L.P.

3–7. SLIDE: Configuring IP Multiplexing



Student Notes

HP-UX version 11.00 introduced “IP Multiplexing” to its TCP/IP protocol stack. This new functionality makes it possible to assign multiple IP addresses to a single physical interface card.

The example on the slide shows one application of this feature. The web server shown in the graphic has a single physical interface card connected to the Internet. However, this single physical interface card has three different “logical” interfaces. Each logical interface has a different IP address, each associated with a different host name and a different instance of the WWW server software. This makes it possible for a server with a single LAN card to host multiple web sites with different IP addresses and host names.

Interface Names in a Multiplexed Environment

Traditionally, HP-UX identified LAN interface cards with simple interface names following the format lan0, lan1, lan2, etc. These interface names were assigned by the system and could be viewed via the **`lanscan`** command.

In a multiplexed environment, a single physical interface may have several logical interfaces. Each logical interface is identified by an index number appended to the physical LAN interface name.

© 2003 Hewlett-Packard Development Company, L.P.

The first index assigned to an interface card is always "0", resulting in logical interface name **lan0:0** (or simply **lan0**). Once you have configured **lan0:0**, subsequent index numbers may be assigned in any order desired. The physical interface card shown on the slide has three logical interfaces configured: **lan0:0**, **lan0:1**, and **lan0:2**. Each logical instance may be assigned a different IP address, and a different host name.

Using IP Multiplexing to Configure IP/Ethernet Versus IP/IEEE 802.3

Logical interfaces are also used when an interface card is used for both IP/Ethernet and IP/IEEE 802.3 frames. You may have noticed two interface names for each LAN card in your **lanscan** output: **lan0** and **snap0**. All HP Ethernet interface cards support both the Ethernet and the IEEE 802.3 encapsulation standards. The interface name you choose to configure determines which encapsulation method will be used. Using the **lan0** interface name ensures that Ethernet encapsulation should be used. Using the **snap0** interface name ensures that the IEEE 802.3 encapsulation standard is used.

A card may be configured to support both encapsulation methods simultaneously by configuring IP addresses for both **lan0** and **snap0**. **lan0** and **snap0** must have different IP addresses, and the two IP addresses must be on different subnets. To provide IEEE 802.3 encapsulation via the LAN card shown on the slide, one would simply add the following three lines to the system's **/etc/rc.config.d/netconf** file:

```
INTERFACE_NAME[3]=snap0:0
IP_ADDRESS[3]=128.4.1.1
SUBNET_MASK[3]=255.255.0.0
```

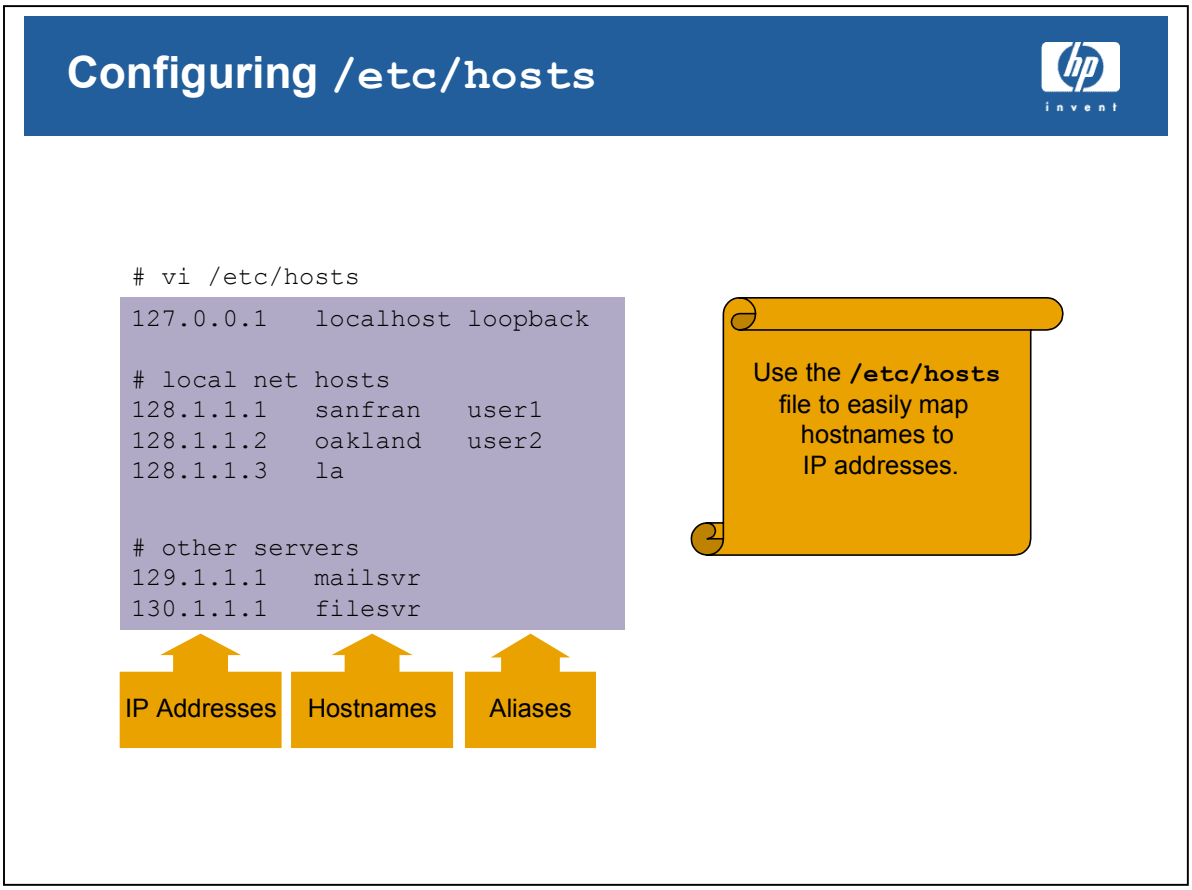
The following **ifconfig** command would execute automatically at boot time as a result of the lines shown above:

```
# ifconfig snap0:0 128.4.1.1 netmask 255.255.0.0 up
```

NOTE:	Each logical interface must have a unique IP address. Logical interfaces that use the same encapsulation method may have IPs on the same subnet. Logical interfaces that use different encapsulation methods, however, must be on different subnets.

© 2003 Hewlett-Packard Development Company, L.P.

3–8. SLIDE: Configuring /etc/hosts



Student Notes

The **/etc/hosts** file is one of several mechanisms HP-UX hosts use to resolve host names into IP addresses. Each **/etc/hosts** file entry must have an IP address and an associated host name. Each entry may also contain one or more optional host name aliases, and an optional comment preceded by a "**#**" sign.

At a minimum your **/etc/hosts** file should contain entries for:

- Each IP address listed in **/etc/rc.config.d/netconf**.
- The 127.0.0.1 loopback address.

Additional entries may be added or modified using **vi**, or any other editor.

- Fields can have any number of blanks or tabs separating them.
- There should be only one host entry per line.
- Do not include leading zeroes in IP addresses.
- Do not change or delete the localhost/loopback line.

© 2003 Hewlett-Packard Development Company, L.P.

NOTE: The **/etc/hosts** file should be owned by **bin** and should have **0444** (**-r--r--r--**) access permission.

Other Name Resolution Mechanisms

The **/etc/hosts** file is just one of several mechanisms available for resolving host names in HP-UX. Your system may be configured to use the Domain Name Service (DNS), Network Information Service (NIS), or NIS+ in conjunction with or as a replacement for **/etc/hosts**. HP-UX consults the **/etc/nsswitch.conf** file to determine which service should be used for name resolution. **/etc/nsswitch.conf** will be discussed later in the course when DNS and NIS are introduced.

© 2003 Hewlett-Packard Development Company, L.P.

3–9. LAB: Configuring Network Connectivity

Directions

This lab will configure a new host name and IP address for each system in your classroom. Please check with the instructor for an assigned host name and IP address.

Preliminary Step

Just in case something goes wrong during this lab, make a backup copy of all of your network configuration files. There is a shell script in your **`labs`** directory designed specifically for this purpose. The shell script will save a tar archive backup of your network configuration files in the file you specify. Run the shell script by typing:

```
# /labs/netfiles.sh -s ORIGINAL
```

Part 1: Checking the Current LAN Card Configuration

Check the current configuration of the LAN card. Answer the following questions related to its configuration.

1. How many LAN cards does your system have, and what are their Hardware paths?

2. Verify that the "Networking" product is installed on your machine. Is any additional networking software installed on your machine to support your LAN interface cards?

3. Does your kernel contain the drivers necessary to support your LAN cards? Which command will tell you if your LAN cards have been CLAIMED by a driver? If your LAN card is UNCLAIMED, install the necessary drivers.

© 2003 Hewlett-Packard Development Company, L.P.

4. Do device files exist for your LAN cards?

5. List the current MAC address, IP address, netmask, and broadcast address for each of your LAN cards.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring the New LAN Card Configuration

The goal of this portion of the lab exercise is to configure a new IP address and host name for each of the machines in the classroom. Your instructor will assign you a host name/IP from the table that follows. All of the addresses listed are on the 128.1.0.0/16 network.

```
corp         128.1.0.1
sanfran      128.1.1.1
oakland      128.1.1.2
la           128.1.1.3

chicago      128.1.2.1
peoria       128.1.2.2
rockford     128.1.2.3

atlanta      128.1.3.1
athens       128.1.3.2
macon        128.1.3.3

nyc          128.1.4.1
albany       128.1.4.2
buffalo      128.1.4.3

paris        128.1.5.1
lyon         128.1.5.2
grenoble     128.1.5.3

london       128.1.6.1
leeds        128.1.6.2
ipswich      128.1.6.3

bonn         128.1.7.1
berlin       128.1.7.2
hamburg      128.1.7.3

tokyo        128.1.8.1
kyoto        128.1.8.2
osaka        128.1.8.3
```

Changing your host name and IP on a running system can wreak havoc on CDE and other applications.

1. Kill CDE before going any further:

```
# /sbin/init.d/dtlogin.rc stop
```


© 2003 Hewlett-Packard Development Company, L.P.

2. Is your lan0 card still "UP" after killing CDE? Look at the "flags" listed in the output from the **`ifconfig`** command.

3. From the command line, change your IP to the address suggested by your instructor from the table above. Also set your netmask to a value appropriate for a /16 network.

4. Is your new IP address set properly? How can you find out?

5. Modify the appropriate startup file to make your IP address change permanent. Allow the system to default the broadcast address. Also, permanently change your host name in this startup file.

6. The system keeps sanitized copies of many system configuration files in **`/usr/newconfig/etc`**. Overwrite your current **`/etc/hosts`** file with a copy of **`/usr/newconfig/etc/hosts`**.

7. Add the hosts and IP addresses from the table above to your new **`/etc/hosts`** file.

© 2003 Hewlett-Packard Development Company, L.P.

8. Define a host name alias for each of the host names in your row. Use the first name of the user sitting at each station as the alias.

9. Reboot to see if your changes worked!

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Checking the New Configuration

1. Check your LAN card's IP. Did the configuration work?

2. The **`hostname`** command will display your system host name. Check to ensure that your host name is set properly.

3. Based on your answers to questions 1 and 2 above, what commands did the **`/sbin/init.d/net`** script appear to execute on your behalf during the boot process?

4. Try to **`ping`** the IP address of one of your classmates who has finished rebooting. Does this work?

5. Try to **`ping`** the host name of one of your classmates who has finished rebooting. Does this work?

6. Try to **`ping`** a neighboring machine using the alias you defined in your hosts file. Does this seem to work?

© 2003 Hewlett-Packard Development Company, L.P.

Module 4 — Configuring IP Routing

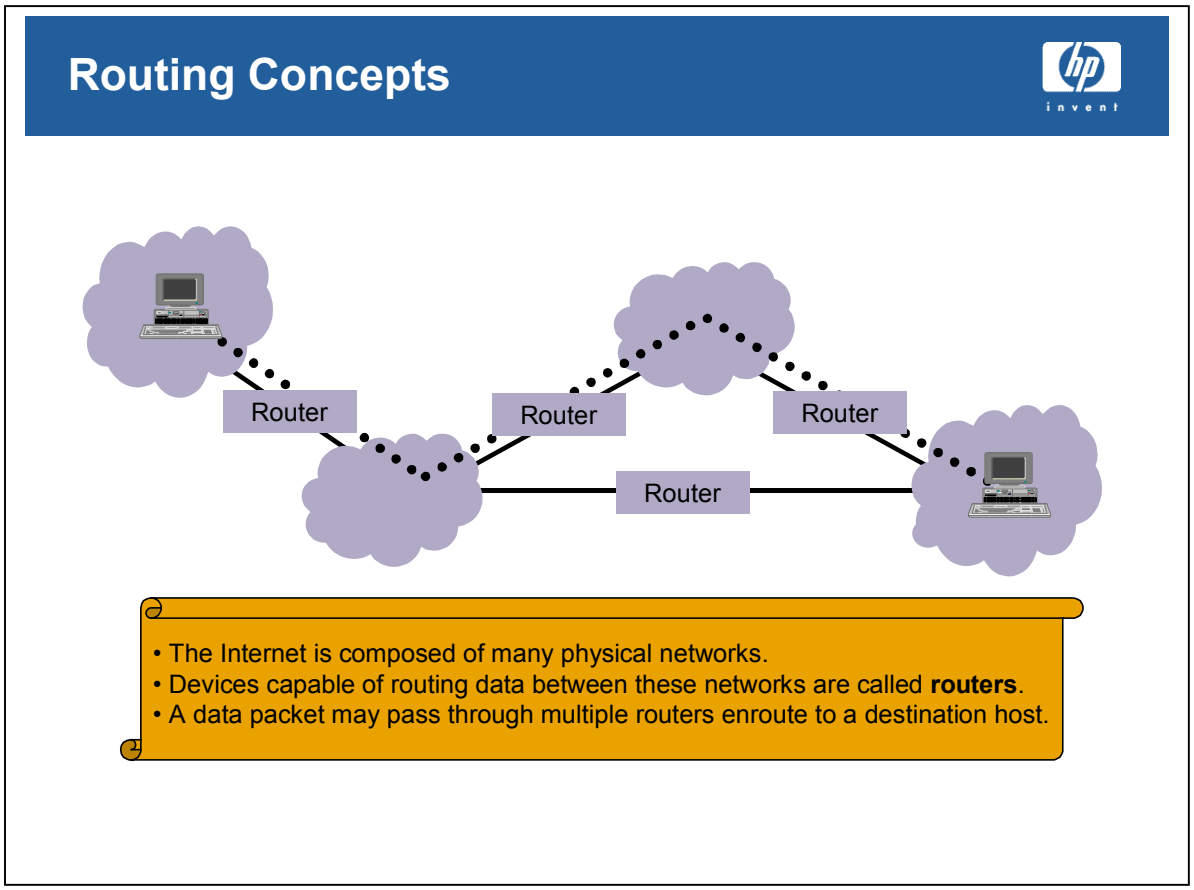
Objectives

Upon completion of this module, you will be able to do the following:

- Configure static routes.
- Configure a default route.
- View the routing tables.

© 2003 Hewlett-Packard Development Company, L.P.

4–1. SLIDE: Routing Concepts



Student Notes

The Internet is composed of many physical networks. Network devices known as routers and gateways interconnect these networks. A network router is a device that is physically connected to two or more networks, and is capable of passing packets between these networks. Any HP 9000 host may be configured as a router, though companies these days more typically use dedicated, specially configured, rack-mounted routers instead.

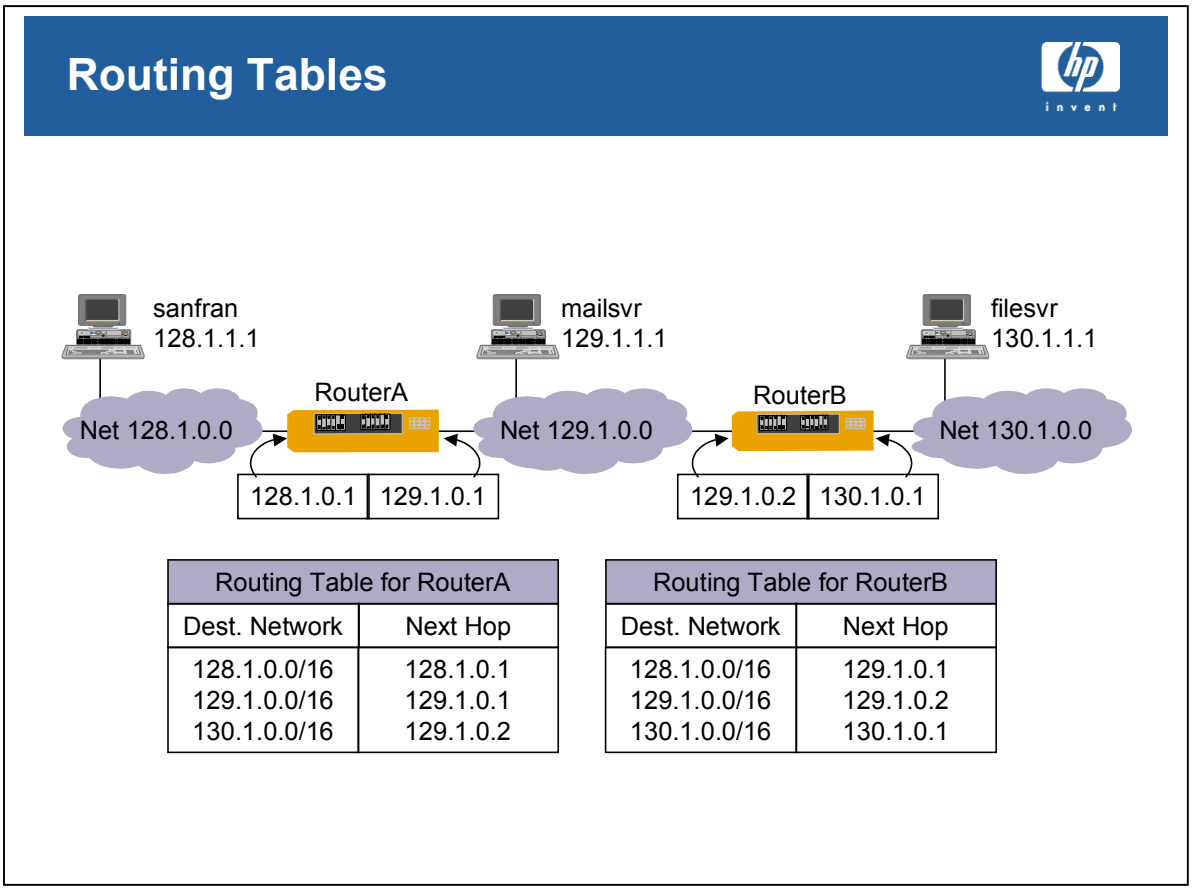
The example on the slide shows several networks interconnected by routers. The host at the top left of the picture wishes to send a packet to the host at bottom right. Since the two hosts are on different networks, the packet must pass through several routers en route to its destination.

The sending host starts by sending the packet to a router on its local network. When the packet reaches the first router, it checks the packet's destination IP to select the next router along the path toward the destination. Packets pass from router to router until they reach a router that can ultimately deliver them directly to the destination host.

IP routing is considered "address-only" routing. This means that packets traveling across the Internet contain only source and destination IP addresses. Along the way, the packet is "told where to turn" by routers.

© 2003 Hewlett-Packard Development Company, L.P.

4–2. SLIDE: Routing Tables



Student Notes

Routers check routing tables maintained in memory to determine where packets should be sent. Each routing table entry contains a pair of addresses.

The first element in each entry identifies a destination network address. When a router receives a packet, it compares the packet's destination IP address to the destination network and addresses in the routing table until a matching entry is identified.

Each routing table entry also identifies the next "hop" required to get to the associated destination network. If the router has a direct connection to the destination network, the "hop" field specifies the IP address of the router LAN card connected to that network. If the router does not have a direct connection to the destination network, the "hop" field identifies the IP address of the next router along the way to that destination.

In either case, the "hop" field must identify an IP address that the router can access directly.

Host-Specific Routes

Although routes are usually defined to entire networks, it is possible to define a route to a specific host. The ability to specify a route for an individual machine is especially useful in troubleshooting.

© 2003 Hewlett-Packard Development Company, L.P.

Examples

The slide shows the routing tables for RouterA and RouterB. However, individual hosts maintain routing tables, too. Complete the routing tables below:

Routing Table for sanfran

Destination	Next Hop
128.1.0.0/16	
129.1.0.0/16	
130.1.0.0/16	

Routing Table for mailsvr:

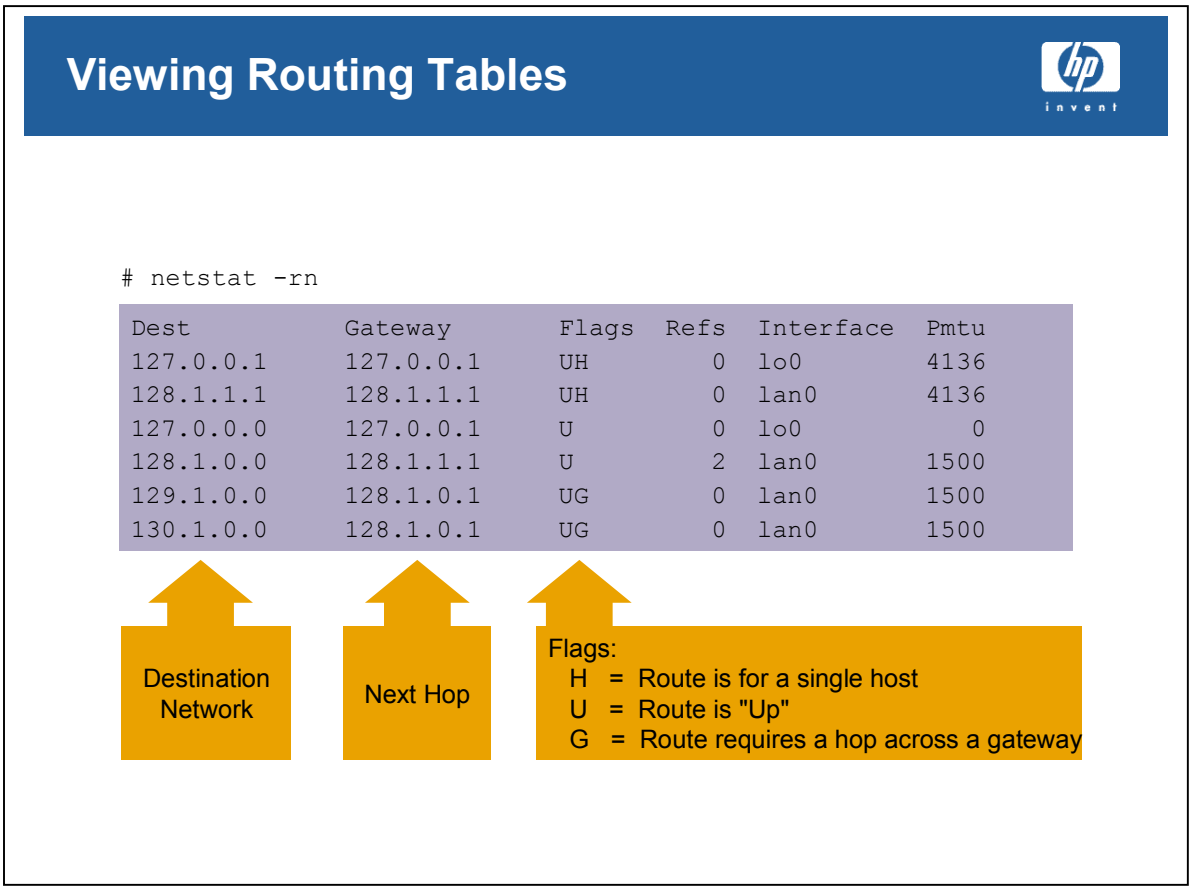
Destination	Next Hop
128.1.0.0/16	
129.1.0.0/16	
130.1.0.0/16	

Routing Table for filesvr:

Destination	Next Hop
128.1.0.0/16	
129.1.0.0/16	
130.1.0.0/16	

© 2003 Hewlett-Packard Development Company, L.P.

4–3. SLIDE: Viewing Routing Tables



Student Notes

You can view your system's routing table via the **`netstat`** command. Each entry in the resulting table includes a "Destination" network or host address, the "Gateway" used to access that destination, and several fields identifying the route usage.

The “Flags” field identifies the following: the route is up (U), the route uses a gateway (G), the destination is a host or network (with or without H), the route was created dynamically (D) by a redirect or by Path MTU Discovery, and a gateway route has been modified (M).

The “Refs” field shows the current number of active uses of the route. Connection-oriented protocols normally use a single route for the duration of a connection, while connectionless protocols obtain a route only while sending a particular message.

The “Interface” field displays the name of the network interface used by the route.

The "Pmtu" field displays the maximum transmission unit size allowed on the interface card used by the route.

© 2003 Hewlett-Packard Development Company, L.P.

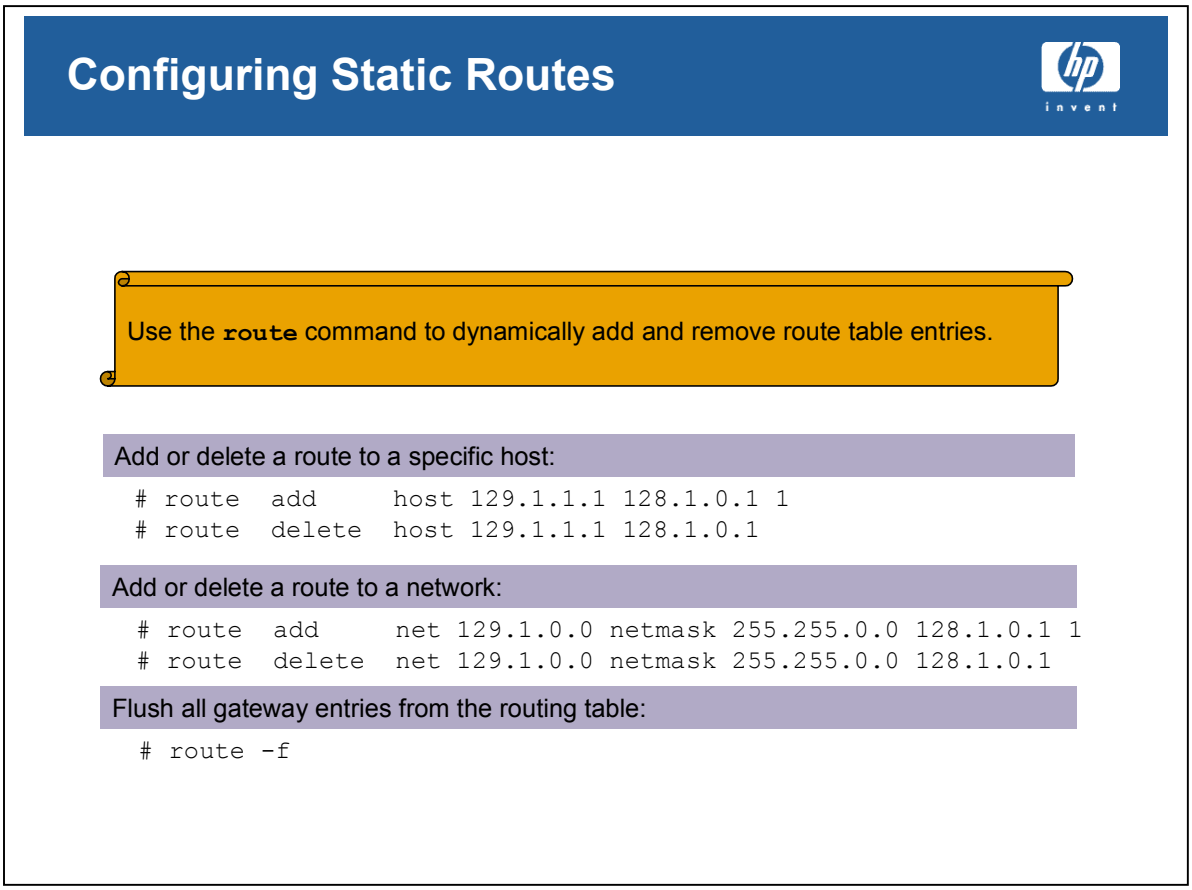
```
# netstat -rn
Dest/Netmask  Gateway       Flags  Refs  Interface  Pmtu
127.0.0.1     127.0.0.1     UH        0  lo0        4136
128.1.1.1     128.1.1.1     UH        0  lan0       1500
127.0.0.0     127.0.0.1     U         0  lo0        4136
128.1.0.0     128.1.1.1     U         2  lan0       1500
129.1.0.0     128.1.0.1     UG        0  lan0       1500
130.1.0.0     128.1.0.1     UG        0  lan0       1500
```

The **-n** option causes **netstat** to display IP addresses rather than host names. If you prefer to view host names in your routing table, leave off the **-n**.

When executed with the **-v** option, **netstat** also displays the netmask associated with each destination in the routing table.

© 2003 Hewlett-Packard Development Company, L.P.

4–4. SLIDE: Configuring Static Routes



Student Notes

You can add and remove entries in your routing table via the **route** command. Consider a few examples.

Adding and Deleting Routes to Individual Hosts

The first couple of examples on the slide add, then delete a route to the host at address 129.1.1.1 via the router at address 128.1.0.1. The ”1” on the end of the command is the “hop count” parameter. This should be set to “0” for hosts on your local network, or “1” if the route requires hops across one or more gateways. The "hop count" is optional when deleting existing routes from the routing table.

```
# route  add     host 129.1.1.1 128.1.0.1 1
# route  delete  host 129.1.1.1 128.1.0.1
```

Adding and Deleting Routes to Entire Networks

Although it is possible to configure routes to individual hosts, it is much more common to configure routes to entire networks. The examples on the slide add, then delete a route to the 129.1.0.0/16 network via the router at address 128.1.0.1. The **netmask** parameter is optional,

© 2003 Hewlett-Packard Development Company, L.P.

but recommended if you are part of a subnetted environment. Here again, the "hop count" indicates if the route requires a hop across a gateway/router.

```
# route  add     net 129.1.0.0 netmask 255.255.0.0 128.1.0.1 1
# route  delete  net 129.1.0.0 netmask 255.255.0.0 128.1.0.1
```

Flushing the Routing Table

The last example flushes all gateway routes from the routing table, leaving nothing but the host's own IP addresses, local routes, and loopback routes. If your routing table becomes corrupted at some point, you may choose to use this option to flush all non-critical routes from the routing table, then re-add the gateway entries manually with the **`route`** command.

```
# route –f
```

Auto-Configured Static Routes

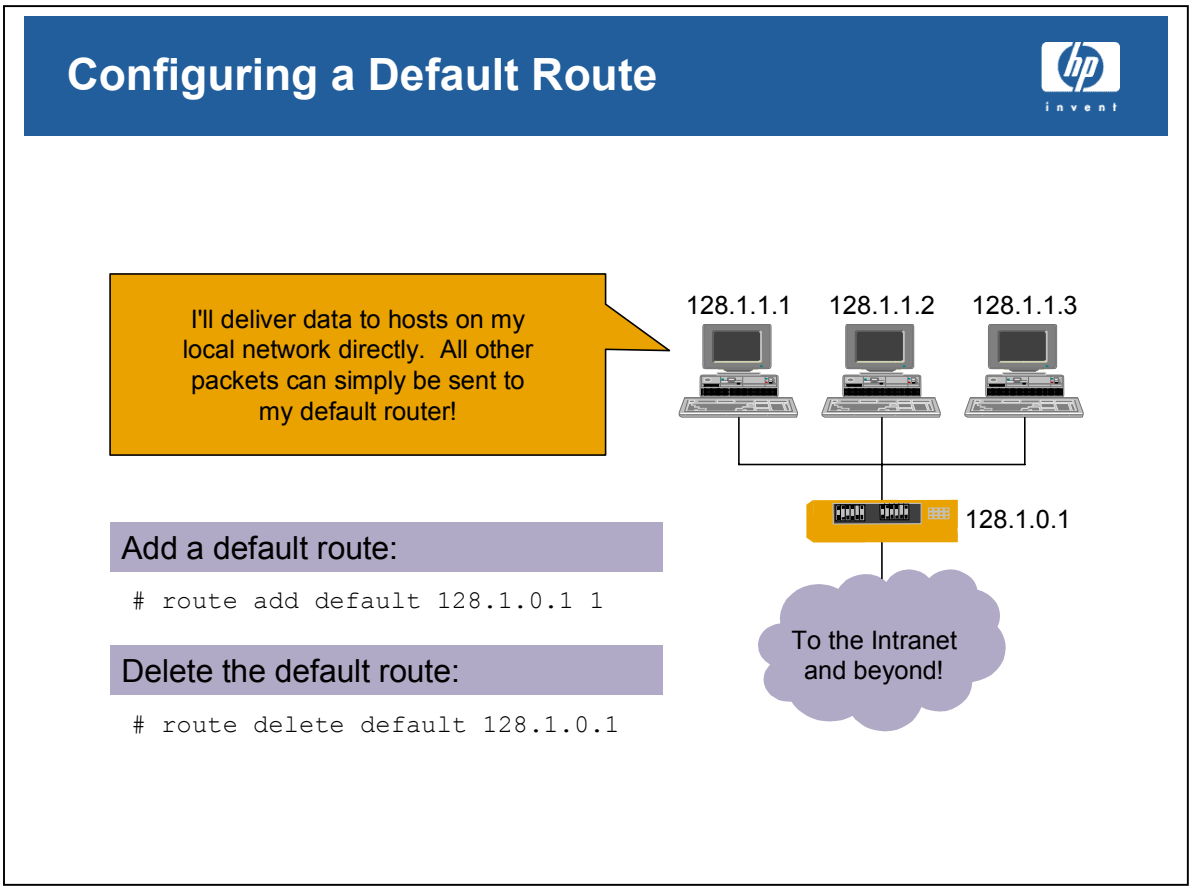
Several routes are configured for you automatically when your IP address and loopback address are set during system startup:

- A route to the host's own IP address.
- A route to the host's own local network.
- A route to the 127.0.0.1 address.
- A route to the 127.0.0.0/8 network.

These four routes must be present in order for your system to function properly!

© 2003 Hewlett-Packard Development Company, L.P.

4–5. SLIDE: Configuring a Default Route



Student Notes

Configuring a Default Router/Gateway

Although an HP-UX workstation or server may be configured as a router, most networks today have dedicated rack-mounted routers. These routers typically support one or more dynamic routing protocols, which continuously exchange information with other routers on the corporate intranet or public Internet. This saves the administrator the drudgery of manually configuring hundreds of entries in the routing tables.

Individual hosts on a network generally maintain routing tables with very few entries. Every host, of course, can directly deliver frames to other hosts on the same network. To reach other networks, most hosts define the nearest dedicated router as the default route in the routing table. The default route is used whenever there is no specified route in the routing table to a destination.

The default route may be defined using the **`route`** command:

```
# route add default 128.1.0.1 1
```


© 2003 Hewlett-Packard Development Company, L.P.

At HP-UX 11.0, it became possible to define multiple default routes on a single host. Defining multiple default routes offers two advantages. First, HP-UX provides some load balancing by sending some packets via the first default router, and others via the second in a round-robin-like fashion. Defining multiple default routes also offers improved reliability. HP-UX monitors the status of the routers; if a router fails to respond, HP-UX uses the alternate default route defined in the routing table.

Configuring Proxy ARP Default Routing

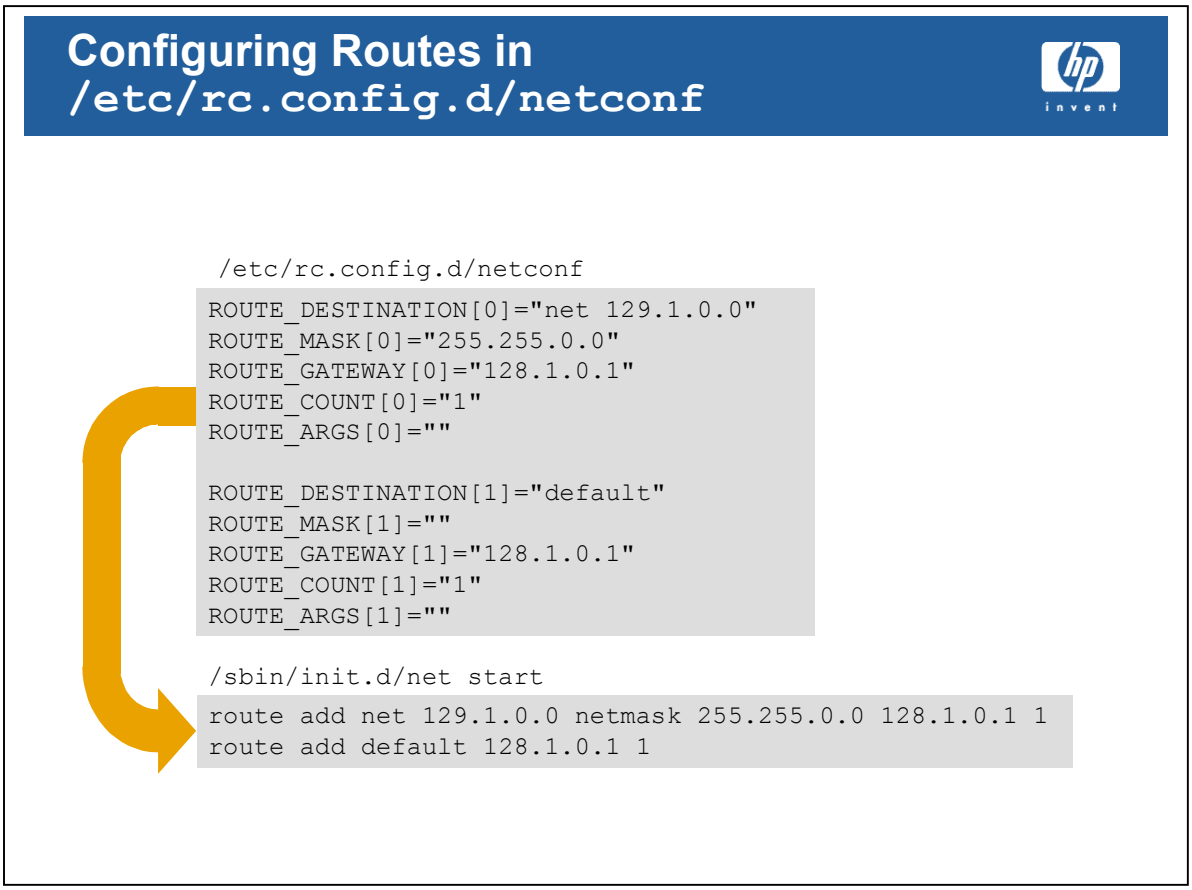
A simpler approach is to simply define your own IP address as the default route. In this case, all packets destined to networks that aren't explicitly listed in your routing table will simply be dropped out on your local network. Assuming your local router supports the "Proxy ARP" protocol, it will recognize all packets destined for IP addresses off the local network and forward them automatically.

The example below configures a proxy ARP default route for host 128.1.1.1. Note that the hop count variable should be null, or set to 0.

```
# route add default 128.1.1.1
```


© 2003 Hewlett-Packard Development Company, L.P.

4–6. SLIDE: Configuring Routes in `/etc/rc.config.d/netconf`



Student Notes

During the system boot process, the **`/sbin/init.d/net`** script consults the **`/etc/rc.config.d/netconf`** file to determine which routes need to be configured. To permanently configure multiple routes, simply replicate the block of **`ROUTE`** variables in the **`netconf`** file, increment the index for each block of lines, and set the variable values accordingly. The slide shows some sample **`netconf`** route entries, and the **`route`** commands that execute as a result of those entries.

You may notice that some of the routes listed in your routing table don't appear in the **`/etc/rc.config.d/netconf`** file. Each time you set or change your IP address, HP-UX automatically creates a route to your own IP and your local network. Similarly, when you remove an IP address, HP-UX automatically removes the route entries associated with that IP address.

The routes to the loopback address (127.0.0.1) and the loopback network (127.0.0.0) are also created automatically.

© 2003 Hewlett-Packard Development Company, L.P.

4–7. LAB: Configuring Routing

Directions

Record the commands you use to perform the tasks suggested below.

Your instructor has configured one of the nodes in your classroom network as a router with two interfaces:

Router's first LAN card IP:	128.1.0.1/16
Router's second LAN card IP:	192.1.1.1/24

Part 1: Viewing and Modifying the Routing Table

1. View your routing table. What routes are currently defined on your host?

2. Are you able to ping the first LAN card on the router? Are you able to **`ping`** the second LAN card on the router? Explain!

3. From the command line, add a route to the 192.1.1.0/24 network. Then check your routing table again to verify that you were successful.

4. Can you **`ping`** the 192.1.1.1 LAN card on **`corp`** now?

© 2003 Hewlett-Packard Development Company, L.P.

5. Delete the 192.1.1.0/24 route table entry. Then check the routing table again to verify that you were successful.

6. Define the router that was configured by your instructor as your default router. Then check your routing table again to be sure this worked.

7. Can you **`ping`** 192.1.1.1 now, even though you do not have an explicit route to 192.1.1.1?

8. How can you ensure that your default route is defined after every system boot? Make it so.

9. Reboot your machine. When your machine comes back up again, check the routing table to verify that the default route is defined.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Adding Router Entries to the `/etc/hosts` File

1. Add an entry to your **`/etc/hosts`** file for **`corp`**'s second LAN card. Since both 128.1.0.1 and 192.1.1.1 are on the same machine, both IP addresses should be mapped to host name **`corp`**.

```
# vi /etc/hosts
   128.1.0.1 corp
   192.1.1.1 corp
```

2. If you **`ping corp`**, which of **`corp`**'s IP addresses does your system appear to choose? Watch your **`ping`** output carefully.

3. For troubleshooting purposes, it may be helpful to be able to specify which IP address is used when pinging a router such as **`corp`**. You may wish to assign **`/etc/hosts`** aliases to each of the LAN cards on **`corp`**.

4. How can you specifically **`ping`** the 192.1.1.1 interface card on **`corp`** now? How can you specifically **`ping`** the 128.1.0.1 interface on "**`corp`**"?

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Important! Backup Your New Network Configuration!

1. Use the **`netfiles`** script to backup the new network configuration that you configured over the last couple of chapters. Many of the labs that follow in this course require access to this archive backup!

```
# /labs/netfiles.sh -s NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 5 — Configuring Subnetting

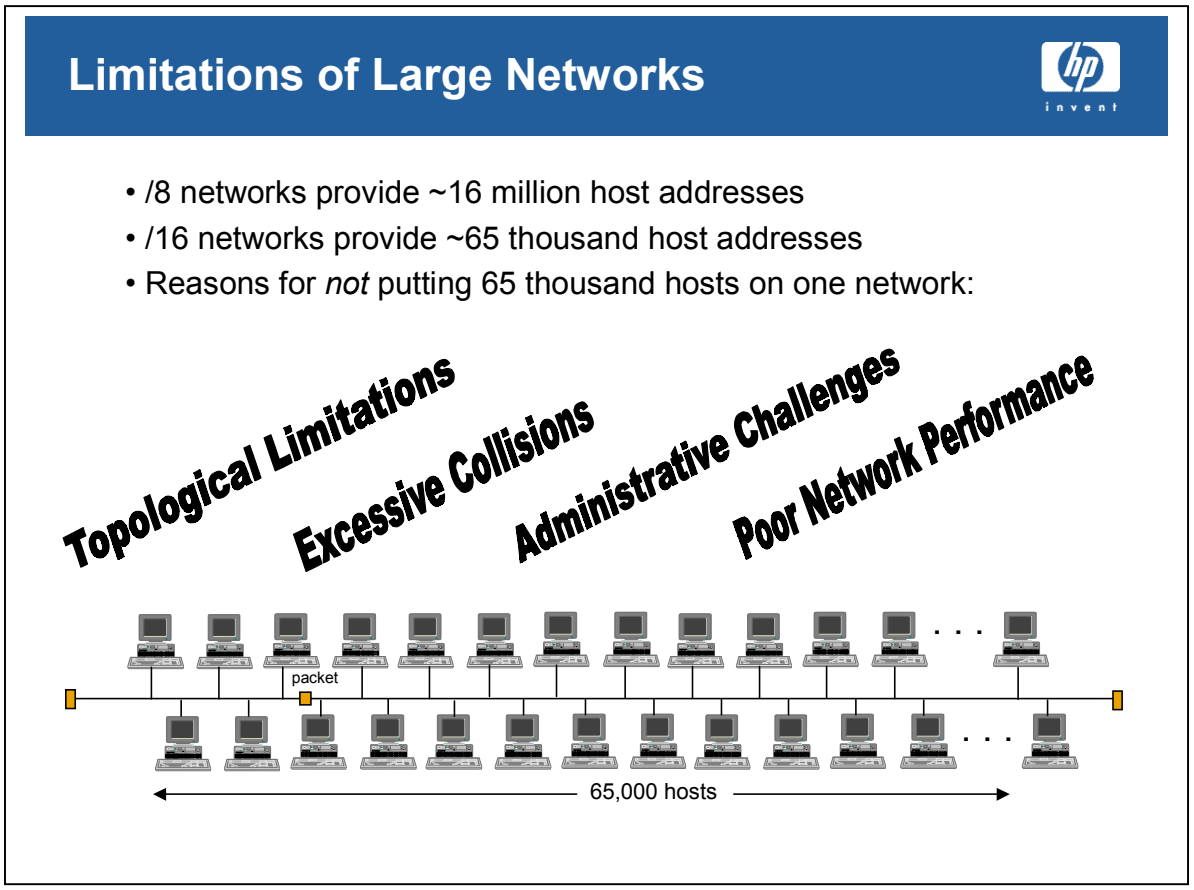
Objectives

Upon completion of this module, you will be able to do the following:

- List the advantages and disadvantages of a subnetted network.
- Subnet a network on an octet boundary.
- Subnet a network on a non-octet boundary.
- Set an HP-UX subnet mask.

© 2003 Hewlett-Packard Development Company, L.P.

5–1. SLIDE: Limitations of Large Networks



Student Notes

Although a /8 network address allows for 16 million host addresses, in reality, it is impractical to have that many hosts sharing a single physical network.

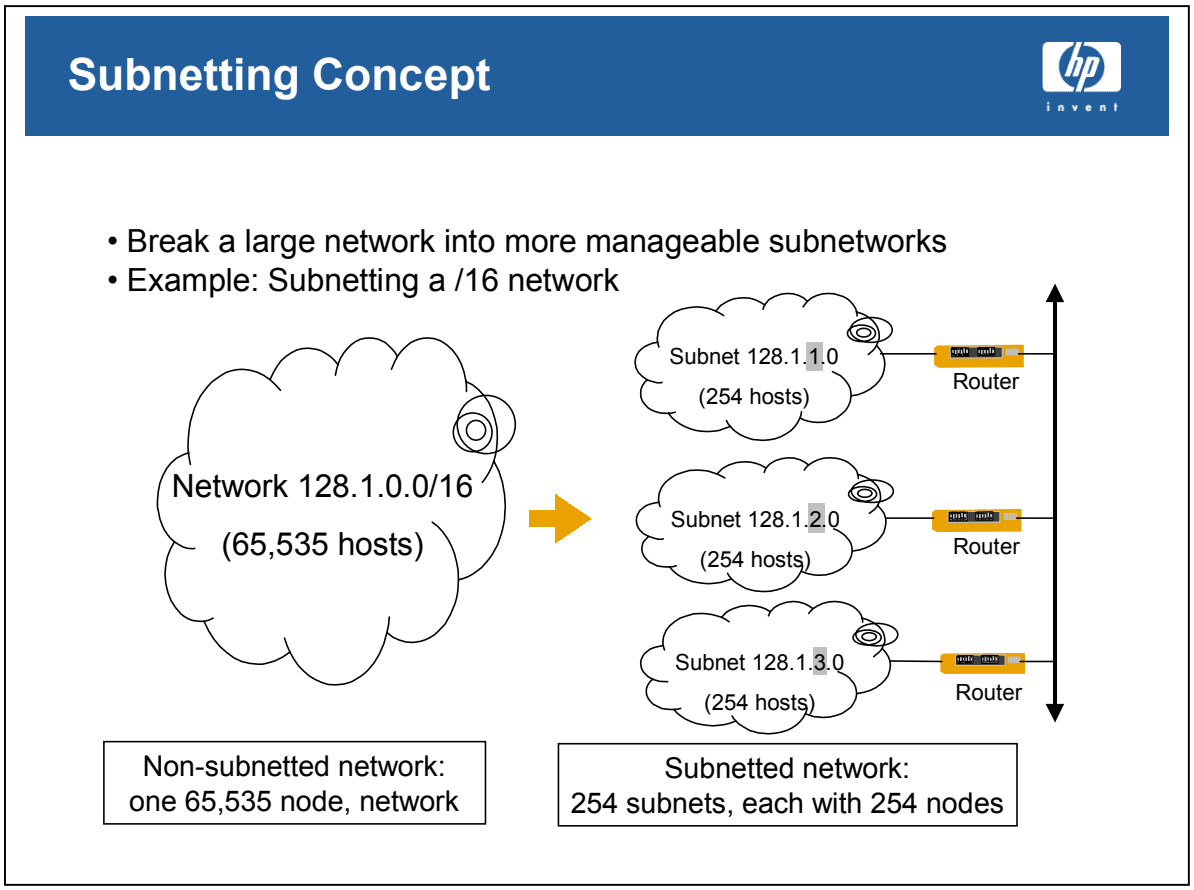
Topological Limitations	Many LAN topologies don't allow 16 million nodes on a single physical network.
Excessive Collisions	If any two nodes on an ethernet network transmit at the same instant, a collision results and both nodes must attempt to retransmit. As the number of nodes on the network increases, the likelihood of collisions increases as well.
Administrative Challenges	Simply keeping track of who has which IP address in a 16-million node network would be an administrative challenge for even the best network administrator.
Poor Network Performance	All of these issues result in degraded network performance as more and more hosts compete for limited bandwidth on a network.

© 2003 Hewlett-Packard Development Company, L.P.

One solution to all of these issues would be to simply leave many of the IP host addresses on /8 networks unused. The rapid depletion of the IP address space however, makes this solution impractical. "Subnetting" provides a much better solution to these problems.

© 2003 Hewlett-Packard Development Company, L.P.

5–2. SLIDE: Subnetting Concept



Student Notes

Subnetting makes it possible to divide a large network IP address space into several smaller, more manageable "subnets."

The example on the slide shows a subnetted /16 network. Without subnetting, the 128.1.0.0/16 network would have 65 thousand hosts on the same physical network, which could easily lead to excessive collisions.

This network, however, has been subdivided into 254 subnets. Each of these subnets could potentially have up to 254 hosts.

```
Subnet Addresses
----------------
128.1.1.0
128.1.2.0
...
128.1.253.0
128.1.254.0
```

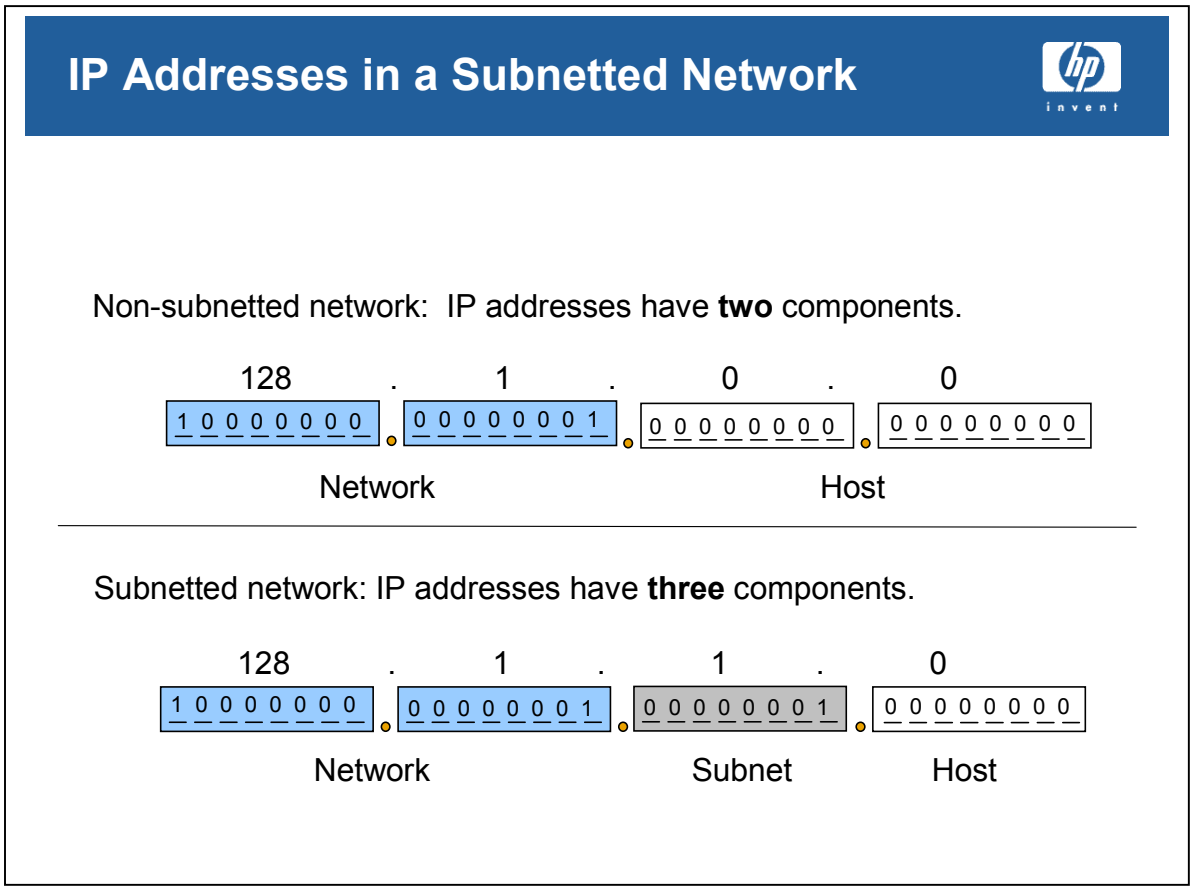
Subnets are separated from one another by routers, which overcome both the collision and topological issues discussed on the previous slide.

© 2003 Hewlett-Packard Development Company, L.P.

Subnetting also makes it easy for the network administrator to delegate authority for portions of the IP network address space to other entities within the organization. Simply assign each department a separate subnet. Each network administrator then becomes responsible for a subnet within the larger corporate network.

© 2003 Hewlett-Packard Development Company, L.P.

5–3. SLIDE: IP Addresses in a Subnetted Network



Student Notes

In a non-subnetted network, each IP address has just two components. A portion of the IP's bits identifies the network to which a host is attached, and the remaining bits uniquely define individual hosts on the network.

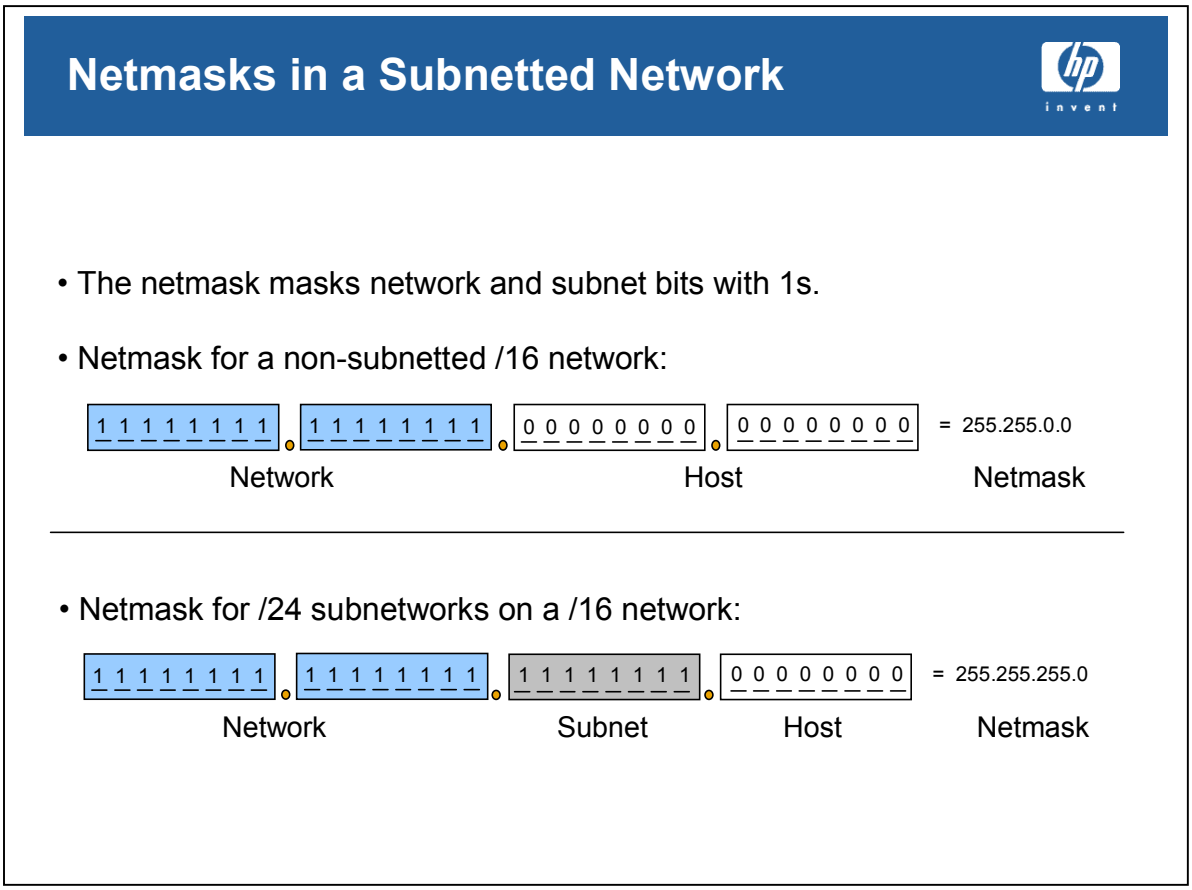
Subnetted IP addresses have a third component as well: a portion of the IP address's host bits is used to define the subnet to which the host belongs.

Returning to the 128.1.0.0/16 network example: Normally, a host on a /16 network has 16 host bits. When implementing subnetting, 8 of those bits are used to define the host's subnet, leaving 8 remaining bits to define the individual host address.

The number of subnet bits may vary. Increasing the number of subnet bits allows more subnets, but fewer hosts on each subnet. Decreasing the number of subnet bits decreases the number of addressable subnets, but allows more hosts on each subnet.

© 2003 Hewlett-Packard Development Company, L.P.

5–4. SLIDE: Netmasks in a Subnetted Network



Student Notes

The text on the previous page noted that the number of subnet bits can vary. So how do routers and other network devices determine where the network/subnet portion of an IP address ends, and where the host portion of an IP address begins on a subnetted network?

In printed form, the boundary between the network/subnet portion of the IP and the host portion of an IP is typically indicated via the "/" suffix on the end of the IP. The number following the "/" indicates the total number of network/subnet bits. All remaining bits are assumed to be host bits. Consider the example on the bottom of the slide. The IP address in the example has 16 network bits and 8 subnet bits. Since 16+8=24, IP addresses on these subnets would be represented as x.x.x.x/24 addresses.

UNIX identifies the network/ subnet host boundary in an IP address via the IP netmask. On a non-subnetted network, the 1s in the netmask identify network bits. On a subnetted network, the 1s in the netmask mask both network and subnet bits.

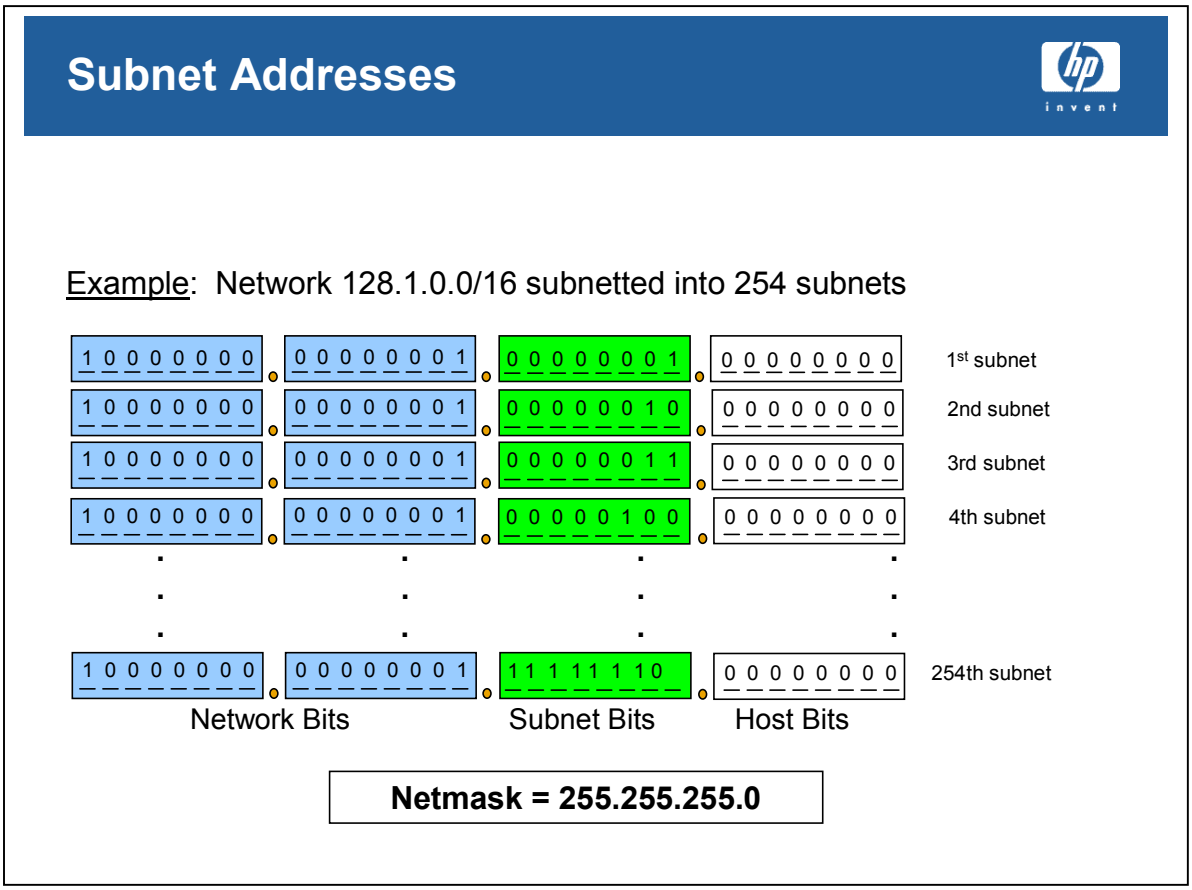
The example on the slide shows a netmask that consists of 24 "1" bits, followed by 8 "0" bits. Thus, the network/subnet portion of the IP addresses on this network appears to span the first three octets, while the final octet represents the host portion of each IP address.

© 2003 Hewlett-Packard Development Company, L.P.

Since the number of subnet bits varies from network to network, the netmask varies from network to network as well. In a subnetted network, you must define the netmask for each LAN interface card.

© 2003 Hewlett-Packard Development Company, L.P.

5–5. SLIDE: Subnet Addresses



Student Notes

A single network may contain multiple subnets. The network bits for all hosts on all of the subnets within a network will be the same. However, each subnet is assigned a unique subnet address. The subnet address is defined in the subnet bits specified by the netmask.

Continuing the example started in the previous slides, this slide shows the subnet addresses for the 128.1.0.0/16 network. The 255.255.255.0 netmask tells us that the third octet defines the subnet portion of the IP addresses on this network.

With eight subnet bits, it is possible to represent 256 addresses:

```
00000000 =   0          Not allowed by some devices.
00000001 =   1
00000010 =   2
00000011 =   3
...
11111101 = 253
11111110 = 254
11111111 = 255          Not allowed by some devices.
```


© 2003 Hewlett-Packard Development Company, L.P.

Although it is possible to represent 256 subnet addresses with 8 subnet bits, some devices and do not allow all-0 or all-1 subnets. Eliminating these addresses leaves the following subnet addresses:

```
128.1.1.0/24
128.1.2.0/24
...
128.1.253.0/24
128.1.254.0/24
```

All-0 and All-1 Subnet Bits in HP-UX

Before HP-UX 11i, HP-UX did not support IP addresses that had all 0s or all 1s in the subnet portion of an IP address. Starting at HP-UX 11i, all-0 and all-1 subnet addresses are supported, but only if the **ip_check_subnet_addr** tunable network parameter has been set to "0". Network tunable parameters, including **ip_check_subnet_addr**, can be both viewed and set using the **ndd** command:

```
# ndd -get /dev/ip ip_check_subnet_addr      Check the current value
# ndd -set /dev/ip ip_check_subnet_addr 0    Enable all-0/all-1 subnets
# ndd -set /dev/ip ip_check_subnet_addr 1    Disable all-0/all-1 subnets
```

By default, this parameter is set to 0, and all-0 and all-1 subnet addresses are allowed.

Changes made via **ndd** are lost at reboot time, unless they are recorded in the **/etc/rc.config.d/nddconf** file:

```
# vi /etc/rc.config.d/nddconf
  TRANSPORT_NAME[1]=ip
  NDD_NAME[1]=ip_check_subnet_addr
  NDD_VALUE[1]=0
```

This is just one of many parameters that may be tuned via the **ndd** command. For a full list of tunable **ndd** parameters, type **ndd -h**.

© 2003 Hewlett-Packard Development Company, L.P.

5–6. SLIDE: Host IP Addresses on a Subnet

Host IP Addresses on a Subnet

- The host address with all 0s represents the address for the entire subnet.
- The host address with all 1s represents the broadcast address for the subnet.
- All other addresses within the subnet may be used for hosts.
- Examples: IP addresses for subnet 128.1.1.0/24:

```
Subnet #1  :  10000000.00000001.00000001.00000000 = 128.1.1.0/24
Host #1    :  10000000.00000001.00000001.00000001 = 128.1.1.1/24
Host #2    :  10000000.00000001.00000001.00000010 = 128.1.1.2/24
Host #3    :  10000000.00000001.00000001.00000011 = 128.1.1.3/24
  :                                  :                    :

Host #253  :  10000000.00000001.00000001.11111101 = 128.1.1.253/24
Host #254  :  10000000.00000001.00000001.11111110 = 128.1.1.254/24
Broadcast  :  10000000.00000001.00000001.11111111 = 128.1.1.255
```

Netmask = 255.255.255.0

Student Notes

Each subnet may contain multiple hosts. Within a subnet, all network and subnet bits must be identical for every host. However, each host must have a unique sequence of host bits to distinguish it from all the other hosts on the subnet.

Consider the 128.1.1.0/24 subnet from the previous page. Each host on this subnet will have an IP address that begins with 128.1.1. This leaves eight host bits.

With eight bits, it is possible to represent 256 values:

```
00000000 =   0
00000001 =   1
00000010 =   2
00000011 =   3
...
11111101 = 253
11111110 = 254
11111111 = 255
```

The address formed by setting all the host bits to 0 is used to define routes to the subnet in the network routing tables. This address should not be assigned to a specific node.

© 2003 Hewlett-Packard Development Company, L.P.

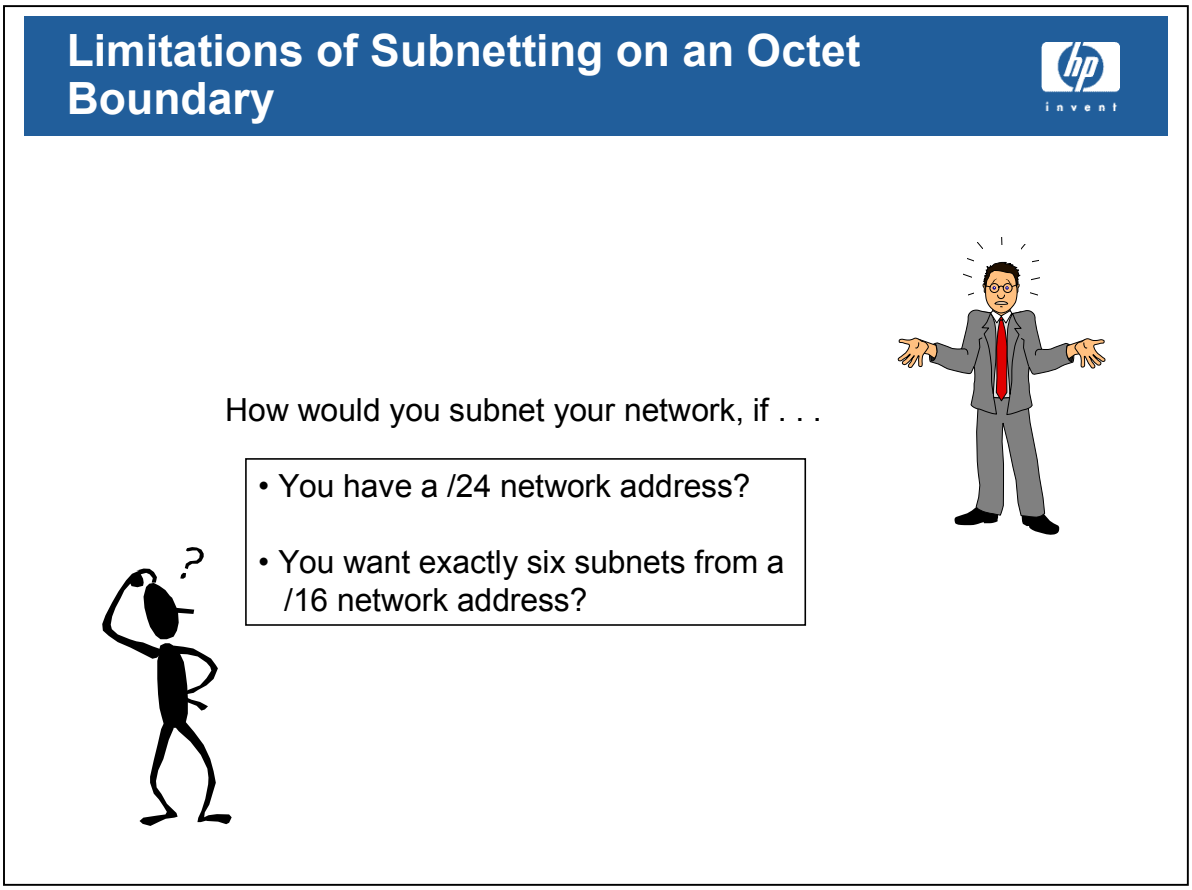
The address formed by setting all the host bits to 1 is a reserved address as well. It is the subnet broadcast address.

All remaining addresses may be assigned to hosts in the subnet. Valid addresses for hosts on the 128.1.1.0/24 subnet, then, include:

```
128.1.1.1/24
128.1.1.2/24
128.1.1.3/24
...
128.1.1.253/24
128.1.1.254/24
```


© 2003 Hewlett-Packard Development Company, L.P.

5–7. SLIDE: Limitations of Subnetting on an Octet Boundary



Student Notes

The example discussed thus far in the chapter used a simple netmask that placed the subnet/host boundary on an octet boundary. Although this makes it easy to determine which subnet a given IP address is on, subnetting on an octet boundary may not provide the flexibility you need as you design your subnets.

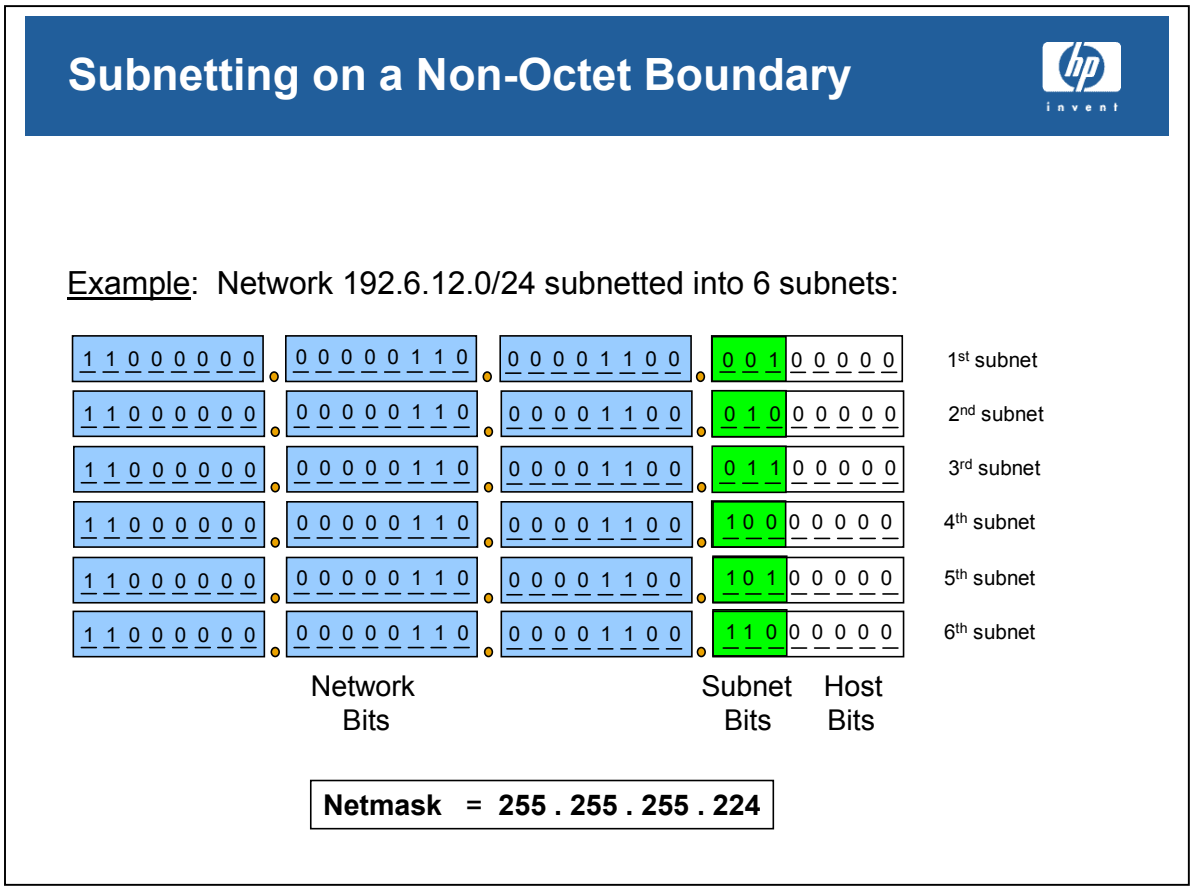
Octet-boundary subnetting is not even an option in a /24 network. Since /24 addresses have just one host octet, using that octet to define an IP's subnet would not leave any host bits!

Octet boundary subnetting may prove limiting on a /16 network, too. What happens if you have a /16 network, and need exactly six subnets? Octet-boundary subnetting would break your network into 254 subnets. This is many more than you actually need.

For these reasons, octet-boundary subnetting rarely offers the flexibility needed to subnet a large network.

© 2003 Hewlett-Packard Development Company, L.P.

5–8. SLIDE: Subnetting on a Non-Octet Boundary



Student Notes

Subnetting on a non-octet boundary simply means that the subnet/host boundary does not fall on an octet boundary. The example on the slide shows a /24 network, 192.6.12.

Formulating the Subnet Address

The administrator has chosen to break the network shown on the slide into six subnets by using three bits from the fourth octet as subnet bits. With three bits, it is possible to represent eight values:

```
000          Not allowed by some routers.
001
010
011
100
101
110
111          Not allowed by some routers.
```

Recall that the subnet address is defined by setting all of the remaining host bits to 0. Thus, the subnet addresses on this network are:

© 2003 Hewlett-Packard Development Company, L.P.

```
192.6.12.001 00000 = 192.6.12.32
192.6.12.010 00000 = 192.6.12.64
192.6.12.011 00000 = 192.6.12.96
192.6.12.100 00000 = 192.6.12.128
192.6.12.101 00000 = 192.6.12.160
192.6.12.110 00000 = 192.6.12.192
```

Formulating the Netmask

The netmask is defined by setting all of the network and subnet bits to 1. In this case the result is:

```
11111111.11111111.11111111.11100000 = 255.255.255.224
```

Formulating the Host Addresses

Taking three bits from the last octet to define the subnet leaves just five bits to define the host portion of the IP. The chart on the text page that follows shows the valid addresses for each subnet.

Recall that the broadcast address for a subnet is formulated by setting all the host bits to 1. The subnet address is formulated by setting all the host bits to 0.

© 2003 Hewlett-Packard Development Company, L.P.

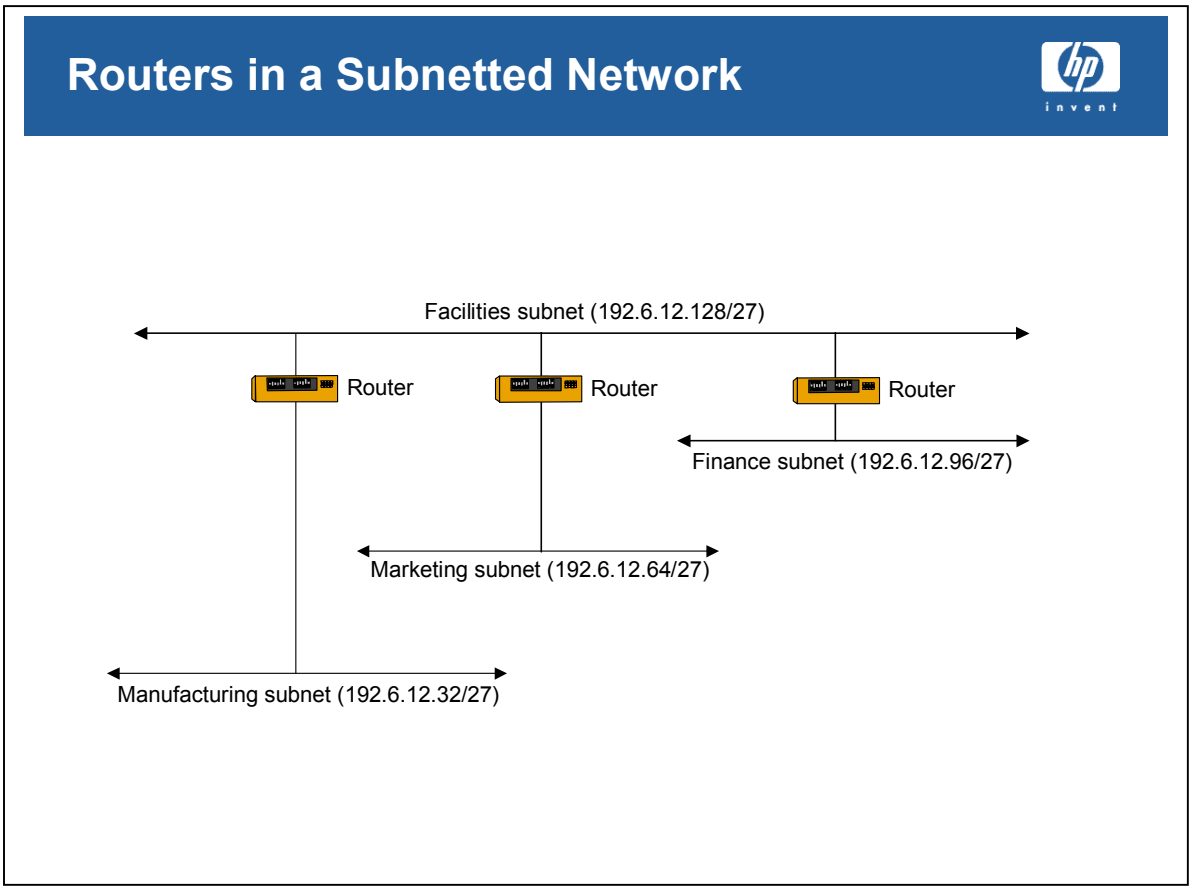
5–9. TEXT PAGE: More Subnetting on a Non-Octet Boundary

The chart below shows all of the IP addresses for the 192.6.12.0/16 network example from the previous page:

IP Address (Decimal & Binary)				IP Address	Usage
192	6	12	00000000	192.6.12.0/24	Network address
192 192 192 192	6 6 6 6	12 12 12 12	00100000 00100001 00111110 00111111	192.6.12.32/27 192.6.12.33/27 192.6.12.62/27 192.6.12.63/27	Subnet #1 Subnet #1, First Host Subnet #1, Last Host Subnet #1, Broadcast
192 192 192 192	6 6 6 6	12 12 12 12	01000000 01000001 01011110 01011111	192.6.12.64/27 192.6.12.65/27 192.6.12.94/27 192.6.12.95/27	Subnet #2 Subnet #2, First Host Subnet #2, Last Host Subnet #2, Broadcast
192 192 192 192	6 6 6 6	12 12 12 12	01100000 01100001 01111110 01111111	192.6.12.96/27 192.6.12.97/27 192.6.12.126/27 192.6.12.127/27	Subnet #3 Subnet $3, First Host Subnet #3, Last Host Subnet #3, Broadcast
192 192 192 192	6 6 6 6	12 12 12 12	10000000 10000001 10011110 10011111	192.6.12.128/27 192.6.12.129/27 192.6.12.158/27 192.6.12.159/27	Subnet #4 Subnet #4, First Host Subnet #4, Last Host Subnet #4, Broadcast
192 192 192 192	6 6 6 6	12 12 12 12	10100000 10100001 10111110 10111111	192.6.12.160/27 192.6.12.161/27 192.6.12.190/27 192.6.12.191/27	Subnet #5 Subnet #5, First Host Subnet #5, Last Host Subnet #5, Broadcast
192 192 192 192	6 6 6 6	12 12 12 12	11000000 11000001 11011110 11011111	192.6.12.192/27 192.6.12.193/27 192.6.12.222/27 192.6.12.223/27	Subnet #6 Subnet #6, First Host Subnet #6, Last Host Subnet #6, Broadcast
255	255	255	11100000	255.255.255.224	Netmask

© 2003 Hewlett-Packard Development Company, L.P.

5–10. SLIDE: Routers in a Subnetted Network



Student Notes

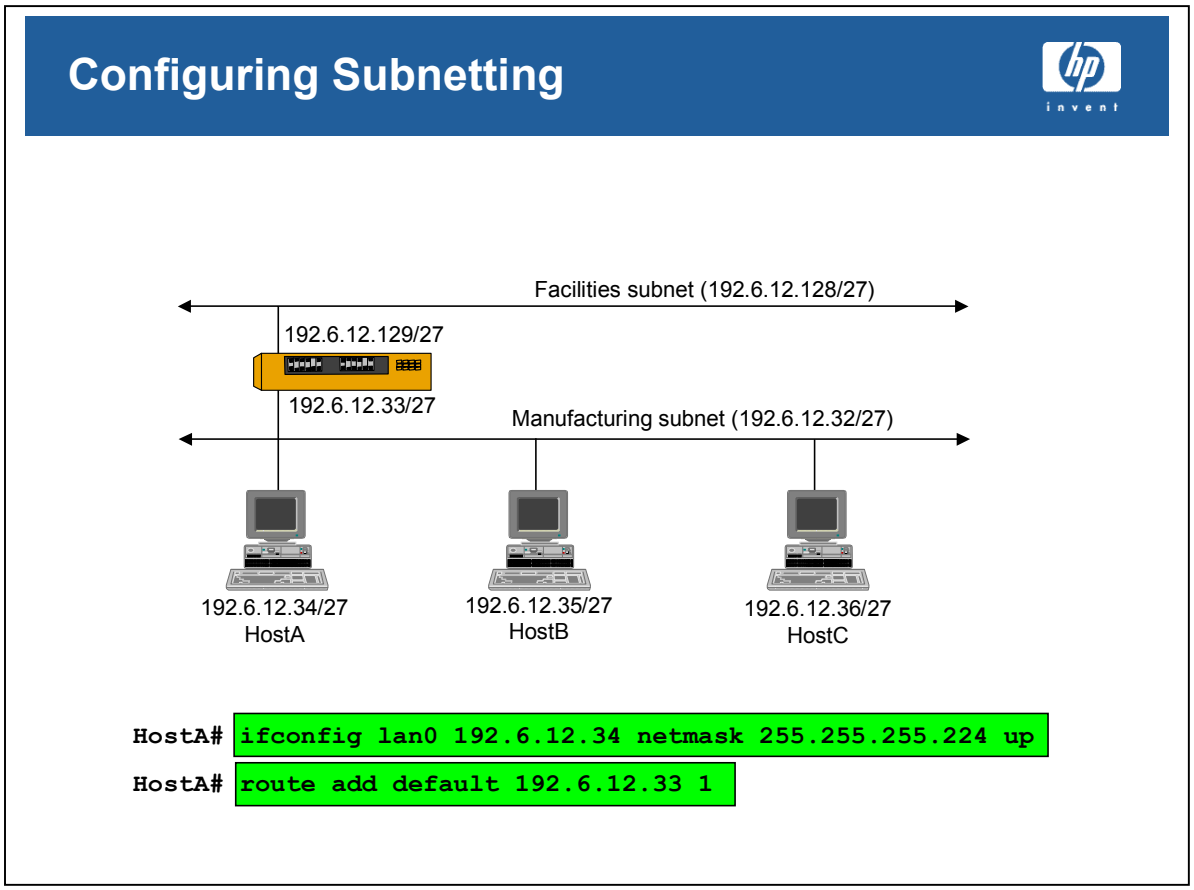
Subnets on the network are separated by routers. In the example on the slide, the facilities subnet is the network backbone. The other three subnets all connect to the facilities subnet via routers.

Although each subnet has a different subnet address, all share the same netmask.

The next slide describes the steps required to configure subnetting of the hosts on the "manufacturing" subnet.

© 2003 Hewlett-Packard Development Company, L.P.

5–11. SLIDE: Configuring Subnetting



Student Notes

This slide shows the steps required to configure subnetting on each of the hosts on the manufacturing subnet. When configuring the interface card on a host connected to a subnetted network, you must specify the subnet mask as an argument to the **`ifconfig`** command. All of the hosts on the subnet must have the same subnet mask.

To ensure that your host has access to other subnets and networks, define a default route to your nearest router. If you wish to make your configuration permanent, modify **`/etc/rc.config.d/netconf`**. For HostA, the **`netconf`** file should contain the following:

```
HOSTNAME=HostA

IP_ADDRESS[0]=192.6.12.34
SUBNET_MASK[0]=255.255.255.224
INTERFACE_NAME[0]=lan0

ROUTE_DESTINATION[0]=default
ROUTE_GATEWAY[0]=192.6.12.33
ROUTE_COUNT[0]=1
```


© 2003 Hewlett-Packard Development Company, L.P.

The **`/etc/rc.config.d/netconf`** file should be similarly configured on other hosts in the manufacturing subnet, with appropriate host names and IP addresses.

© 2003 Hewlett-Packard Development Company, L.P.

5–12. TEXT PAGE: Class B and Class C Subnetting Reference Sheet

You may use as many of the host bits as you wish to define the subnet portion of an IP. Increasing the number of subnet bits increases the number of subnets available, but decreases the number of hosts on each subnet. The following formulas determine how many subnets and hosts per subnet may be defined, if all-0 and all-1 subnet addresses are not allowed:

$2^{\text{number of subnet bits}} - 2 \geq$ number of subnets available

$2^{\text{number of host bits}} - 2 =$ number of host addresses per subnet

Allowing all-0 and all-1 subnet addresses changes the first formula slightly:

$2^{\text{number of subnet bits}} - 2 =$ numbers of subnets available

$2^{\text{number of host bits}} - 2 =$ number of host addresses per subnet

The tables below show the number of subnets and hosts available for various netmasks on /16 and /24 networks, excluding the all-0 or all-1 subnets.

Net Type	# Subnet Bits	# Host Bits	Netmask	# Subnets	# Hosts
/16	0	16	255.255.0.0	0	65534
	2	14	255.255.192.0	2	16382
	3	13	255.255.224.0	6	8190
	4	12	255.255.240.0	14	4094
	5	11	255.255.248.0	30	2046
	6	10	255.255.252.0	62	1022
	7	9	255.255.254.0	126	510
	8	8	255.255.255.0	254	254
	9	7	255.255.255.128	510	126
	10	6	255.255.255.192	1022	62
	11	5	255.255.255.224	2046	30
	12	4	255.255.255.240	4094	14
	13	3	255.255.255.248	8190	6
	14	2	255.255.255.252	16382	2

Net Type	# Subnet bits	# Host Bits	Netmask	# Subnets	# Hosts
/24	0	8	255.255.255.0	0	254
	2	6	255.255.255.192	2	62
	3	5	255.255.255.224	6	30
	4	4	255.255.255.240	14	14
	5	3	255.255.255.248	30	6
	6	2	255.255.255.252	62	2

© 2003 Hewlett-Packard Development Company, L.P.

5–13. LAB: Configuring Subnets

Directions

Answer all of the questions below. Assume that your network contains some older devices that don't support all-0 or all-1 subnet addresses.

Part 1

1. Your company's network address is 128.20.0.0/16, but your netmask is set to 255.255.255.0. Given this netmask, how many bits are in the subnet portion of your IP address?

2. Given your answer to the previous question, how many host addresses may be configured on each subnet?

3. What are the lowest and highest subnet addresses?

4. What are the lowest and highest host addresses on the first subnet?

© 2003 Hewlett-Packard Development Company, L.P.

Part 2

Your company's network address is 192.30.40.0/24, and you need to create two subnets.

1. How many contiguous bits are needed, and in which octet?

2. What is the subnet mask?

3. What are the valid subnet addresses?

© 2003 Hewlett-Packard Development Company, L.P.

Part 3

Your company's network address is 132.40.0.0/16. You need to configure nine subnetworks.

1. How many bits are needed to form 9 subnets?

2. What will be the subnet mask in dotted decimal notation?

3. List the first three subnet addresses.

4. How many hosts can be on each subnet?

5. What is the complete address for the first host on the first subnet?

6. What would be the complete address for the last host on the first subnet?

© 2003 Hewlett-Packard Development Company, L.P.

7. Fill in the variable values you would expect to see in the **/etc/rc.config.d/netconf** file for the last host on the first subnet. Record the variable values below, but do not actually modify the **/etc/rc.config.d/netconf** file on your system.

```
INTERFACE_NAME[0]=lan0
IP_ADDRESS[0]=
SUBNET_MASK[0]=
```

8. What command would the **/sbin/init.d/net** script execute as a result of the **netconf** values in the previous question?

© 2003 Hewlett-Packard Development Company, L.P.

Module 6 — Troubleshooting Network Connectivity

Objectives

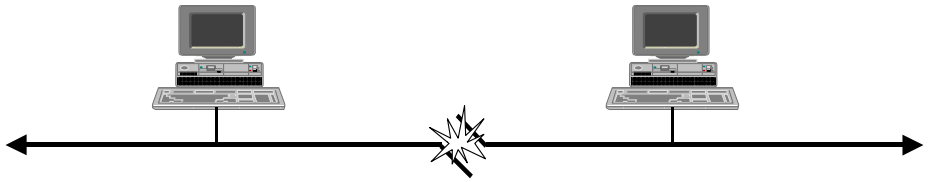
Upon completion of this module, you will be able to do the following:

- Use the following tools to troubleshoot network connectivity:
  - **lanscan**
  - **lanadmin**
  - **linkloop**
  - **arp/ndd**
  - **ping**
  - **netstat -i**
  - **netstat -a**
  - **netstat -r**
  - **hostname**
  - **nslookup**

© 2003 Hewlett-Packard Development Company, L.P.

6–1. SLIDE: Network Troubleshooting Tools Overview

Network Troubleshooting Tools Overview



Several network troubleshooting tools are included with HP-UX, including:

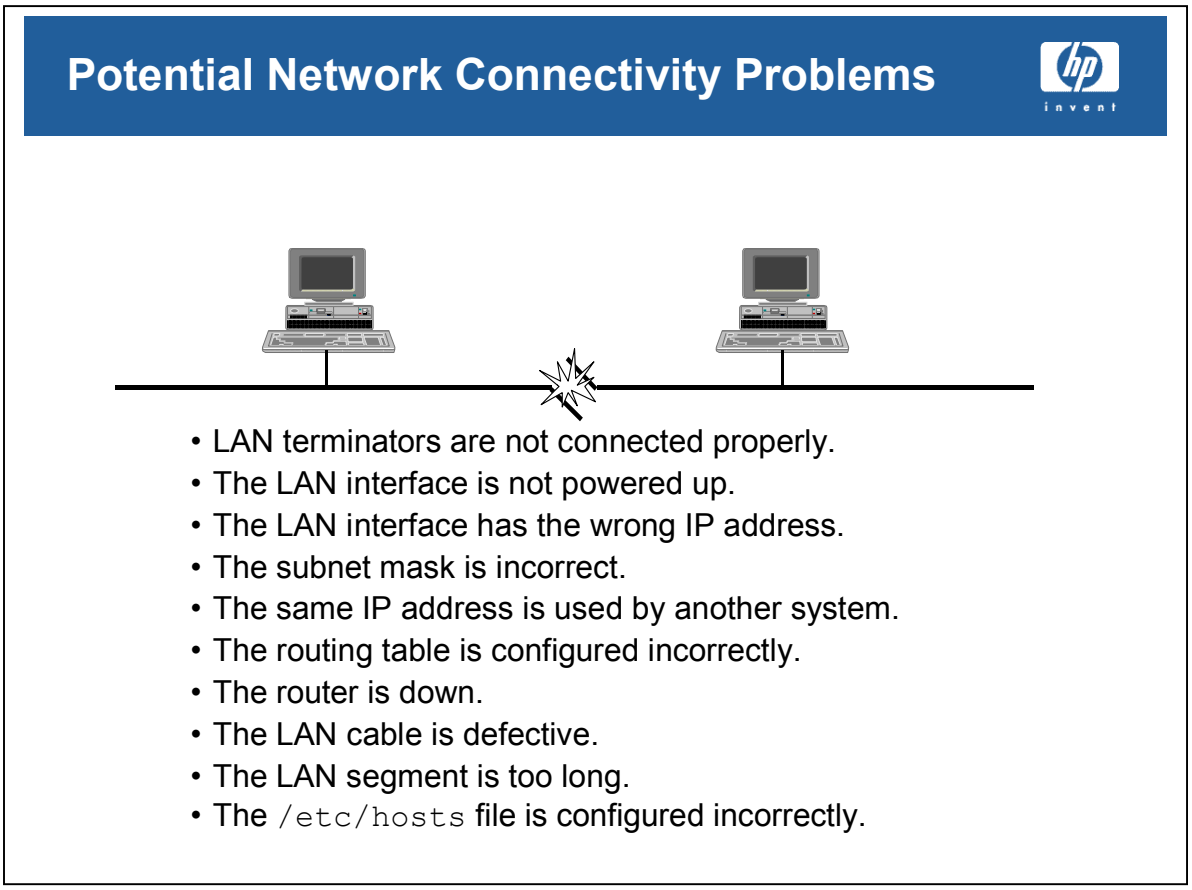
- `lanscan` (HP-specific tool)
- `lanadmin` (HP-specific tool)
- `linkloop` (HP-specific tool)
- `arp` (BSD)
- `ping` (public domain)
- `netstat` (BSD)
- `nslookup` (BSD)

Student Notes

Connectivity problems are not always clearly and directly shown by the tools. Often you get only hints, which you have to interpret. You will have to use several tools in logical steps; therefore, you must be knowledgeable about the networking concepts and the capabilities of each networking tool.

© 2003 Hewlett-Packard Development Company, L.P.

6–2. SLIDE: Potential Network Connectivity Problems



Student Notes

- LAN terminators not connected properly.

  Many times users do not terminate their LAN cables properly. You must have two terminators on your network—one at each end.

- The LAN interface is not powered up.

  The **`ifconfig`** command fails if the LAN interface is defective. You may inadvertently introduce syntax errors into the configuration files if you modify these files with an editor such as **`vi`**.

- The LAN interface has the wrong IP address.

  Someone may have made a mistake when configuring the IP_ADDRESS within the **`/etc/rc.config.d/netconf`** file.

- The subnet mask is incorrect.

© 2003 Hewlett-Packard Development Company, L.P.

Someone may have made a mistake when configuring the SUBNET_MASK within the **/etc/rc.config.d/netconf** file.

- The same IP address is used by another system.

  Sometimes someone connects his or her system to the network without asking the network administrator for a unique IP address.

- The routing table is configured incorrectly.

  Someone may have made a mistake when configuring the ROUTE parameters within the **/etc/rc.config.d/netconf** file.

- The router is down.

  Sometimes a system must be shut down. If you are shutting down a router, you should announce the shutdown at least one day in advance.

- The LAN cable is defective.

  There are specific instruments to detect a break in a cable.

- The LAN segment is too long.

  If coaxial cables were installed a long time ago without using a cabling map, it is possible that the cables have become too long. When a new system is added to the segment, if the cable is extended beyond the segment length limitation, problems will eventually arise. There are cable testers to measure cable lengths.

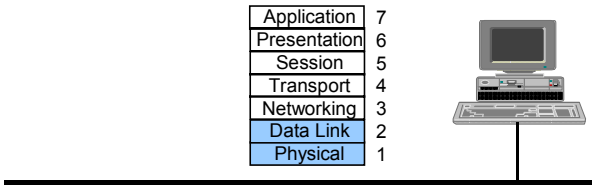
- The **/etc/hosts** file is configured incorrectly.

  If your system cannot resolve a host name to the correct IP address, you probably have a problem in your **hosts** table. When using **/etc/hosts**, the first match working down from the top of the file is used. If two IP addresses are in **/etc/hosts** (for example, for a gateway), **gethostbyname()** will always return the first IP address, which may not be the desired one. You should check your **hosts** file regularly to make sure the entries for your machines are correct.

© 2003 Hewlett-Packard Development Company, L.P.

6–3. SLIDE: The `lanscan` Command

The `lanscan` Command



- The `lanscan` command lists information for all LAN interface cards within the system.

Example (for D-Series system)

```
# lanscan
Hardware Station         Crd Hdw   Net-Interface  NM  MAC       HP-DLPI DLPI
Path     Address         In# State NamePPA        ID  Type      Support Mjr#
8/16/6   0x0060B0A39825  0   UP    lan0 snap0     1   ETHER     Yes     119
8/20/5/1 0x0060B058A8C6  1   UP    lan1 snap1     2   ETHER     Yes     119
```

Student Notes

Any user can execute this simple and quick command. It provides the most efficient way to determine the link level address of the interface card.

It also displays the following information:

Hardware path	HP-UX hardware address of the LAN interface, also displayed by **`ioscan`**.
Station address	Link level address.
Crd IN#	Card instance number, which is a logical number for the hardware path (displayed by **`ioscan -f`**).
Hardware state	Autoconfigured (up) or not autoconfigured (down).
Net-Interface Name PPA	The network interface **Name** and the **PPA** number are concatenated together. A single hardware device may have multiple **NamePPA** identifiers, which indicates multiple encapsulation methods may be supported on the device.

© 2003 Hewlett-Packard Development Company, L.P.

NM ID	Network management ID, which is assigned uniquely by the system. It is used by **lanadmin**, a diagnostic tool.
MAC type	Specifies the medium access control (MAC) standard of the LAN link.
HP DLPI support	Indicates whether or not the LAN device driver will work with HP's Common Data Link Provider interface. It must be yes to use diagnostics **linkloop** and **lanadmin**.
Mjr Num	DLPI major number

Syntax of lanscan

```
/usr/sbin/lanscan [-aimnpv]
```

in which

a Displays station addresses (link level address) only.

i Displays interface names (lan?) only.

m (new in 11.0) Displays MAC types only.

n Displays network management id only.

p (new in 11.0) Displays PPA numbers only.

v Provides verbose output. The output consists of additional lines per interface, and includes the encapsulation method (**IEEE** and/or **ETHER**).

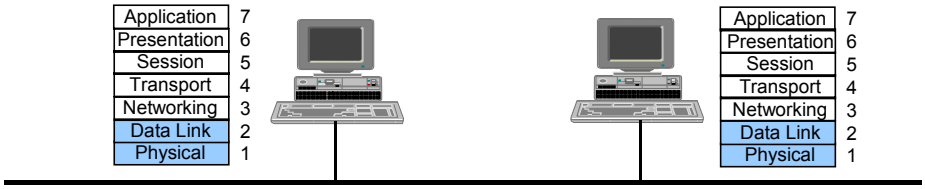
For more information, please see the **man** page **lanscan(1M)**.

NOTE:	Before HP-UX 10.30, **lanscan** displays the interface state of each networking device. This will no longer be the case. LAN drivers no longer maintain the interface state. The Network Interface State field has been removed from the **lanscan** output. Instead, the **netstat** command can be used to determine the state of the interface.

© 2003 Hewlett-Packard Development Company, L.P.

6–4. SLIDE: The `linkloop` Command

The `linkloop` Command



- The `linkloop` command tests layer 2 connectivity.
- The `linkloop` command succeeds even if the destination IP address is not configured or is incorrect.

Example

```
# linkloop 0x0060b007c179
Link connectivity to LAN station: 0x0060b007c179
-- OK
```

Student Notes

`/usr/sbin/linkloop` tests the physical and data link layers (layers 1 and 2) of the OSI model.

`linkloop` uses IEEE 802.3 link test frames to check connectivity within a LAN. You must be root to execute the **`linkloop`** command.

NOTE: **`linkloop`** requires the device file **`/dev/dlpi`** and the **`dlpi`** kernel driver.

The **`linkloop`** command is a quick way to test your own LAN interface. If you provide **`linkloop`** with the link level address of the machine for which you want to test connectivity, **`linkloop`** will report whether or not the connectivity is OK. The link level address can be obtained with the commands **`lanscan`** and **`lanadmin`**.

Before HP-UX 10.30, LAN drivers maintained the interface state. Beginning with HP-UX 10.30, the physical point of attachment (PPA) number for DLPI is no longer equivalent to the network management identifier (NMID). The PPA number has been changed to be the same as the card instance number.

The **`linkloop`** syntax, shown on the slide, has the following parameters:

© 2003 Hewlett-Packard Development Company, L.P.

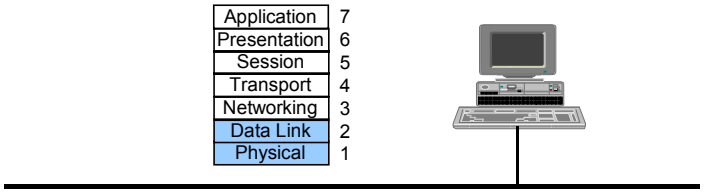
-n *count*	Sets the number of frames to transmit.
-i *PPA*	Specifies the PPA to use. If this option is omitted, **linkloop** uses the first PPA it encounters in an internal data structure. (For releases earlier than HP-UX 10.30, this option will refer to the ***nmid***, which refers to the network management ID as displayed by **lanscan**.)
-t *timeout*	Specifies time in seconds to wait for a reply.
-s *size*	Specifies the size of the data packet.
-v	Verbose option.
linkaddr	The link level address.

For more information, see the **man** page for **linkloop(1M)**.

© 2003 Hewlett-Packard Development Company, L.P.

6–5. SLIDE: The `lanadmin` Command

The `lanadmin` Command



- The `lanadmin` command is a LAN diagnostic tool available specifically for HP 9000 business servers and workstations.
- Capabilities of the `lanadmin` command:
  - resets the LAN interface card
  - changes the maximum packet size for the LAN card
  - changes the speed setting of the LAN card
  - displays driver statistics for the LAN card
  - resets the driver statistics to zero for the LAN card

Student Notes

`lanadmin` allows you to do the following:

- Display and change the station address.
- Display and change the maximum packet size (MTU-max transmission unit) for the LAN card.
- Display and change the maximum speed setting for the LAN card.
- Gather LAN interface statistics.
- Reset the interface card.
- Execute the interface self-test to check for hardware problems.

The following are the **`lanadmin`** command options:

`-e` Echos the input commands on the output device. This is useful if you want to redirect your output to a file.

`-t` Suppresses the display of the command menu before each command prompt. This is the same as the test selection mode **`terse`** command.

© 2003 Hewlett-Packard Development Company, L.P.

-a Display current station address corresponding to PPA Number. The **-A** argument can be used to change the station address.

-m Display current MTU size corresponding to PPA Number. The **-M** argument can be used to change the MTU size.

-s Display current speed setting corresponding to PPA Number. The **-S** argument can be used to change the speed setting.

-h Display on-line help related to the syntax of the command.

When executed in the most common way, without parameters, the following menu is displayed:

```
# /usr/sbin/lanadmin

          LOCAL AREA NETWORK ONLINE ADMINISTRATION, Version 1.0
                        Fri, May 27,1994 16:38:54

                  Copyright 1994 Hewlett Packard Company.
                        All rights are reserved.

Test Selection mode.

      lan     = LAN Interface Administration
      menu    = Display this menu
      quit    = Terminate the Administration
      terse   = Do not display command menu
      verbose = Display command menu

Enter command: lan
```

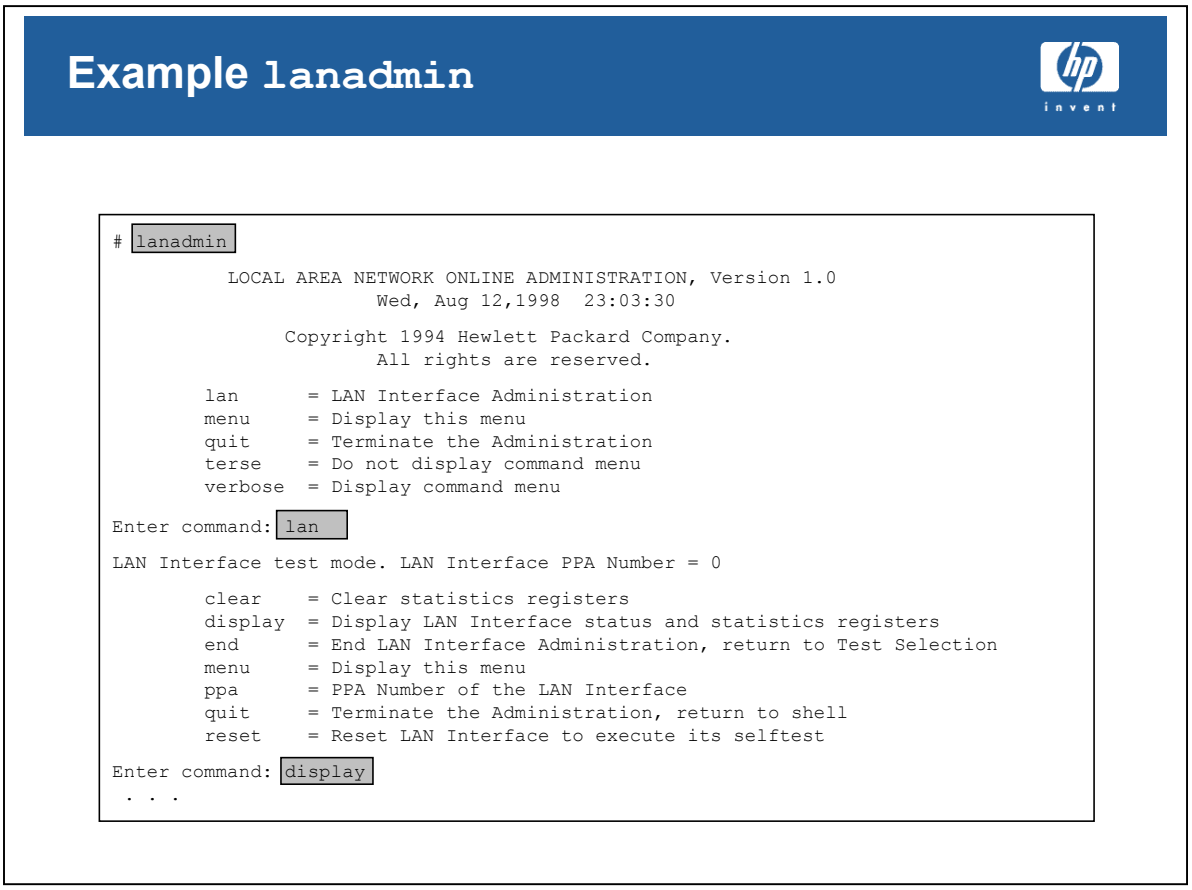
When you invoke **lanadmin**, you are in the test selection mode. From here, you have only one choice. Either enter the diagnostic by entering **lan** or just the first letter, **l**.

The LAN interface diagnostic allows you to test your LAN hardware (layers 1 and 2 of the OSI model).

NOTE: **lanadmin** requires the device file **/dev/dlpi** and the kernel driver **dlpi**.

© 2003 Hewlett-Packard Development Company, L.P.

6–6. SLIDE: Example `lanadmin`



Student Notes

To enter the LAN interface test mode, type **`lan`** while in the test selection mode. The LAN interface test mode allows you to test the physical and data link layers (layers 1 and 2) of the OSI model. Specifically, you can gather LAN interface statistics, reset the interface card, and execute the interface self-test to check for hardware problems.

The following are the LAN interface test commands:

`clear` Clears the LAN interface card network statistics registers to zero. *This command requires superuser status to execute.*

`display` Displays the local LAN interface card status and statistics registers. Allows you to find out how busy the network is.

`end` Returns the diagnostic to the test selection mode.

`menu` Displays the LAN interface test mode command menu.

`ppa` Allows you to tell **`lanadmin`** which interface card to test.

© 2003 Hewlett-Packard Development Company, L.P.

quit Terminates **lanadmin**.

reset Resets the local LAN interface card, causing it to execute its self-test. Local access to the network is interrupted. *This command requires superuser status to execute*. Resetting the card may be necessary when the host has been disconnected from the LAN cable for a long time.

NOTE: If you have a second LAN interface, you must create the proper device files for the interface (for example, **/dev/lan1**) in order to use this diagnostic.

The following is the output from the **display** command.

```
                          LAN INTERFACE STATUS DISPLAY
                           Fri, March 13,1998 16:56:51

PPA Number                      = 0
Description                     = lan0 Hewlett-Packard LAN Interface Hardware Rev 0
Type (value)                    = ethernet-csmacd(6)
MTU Size                        = 1500
Speed                           = 10000000
Station Address                 = 0x80009707445
Administration Status (value)   = up(1)
Operation Status (value)        = up(1)
Last Change                     = 100
Inbound Octets                  = 2887895
Inbound Unicast Packets         = 23560
Inbound Non-Unicast Packets     = 6382
Inbound Discards                = 0
Inbound Errors                  = 833
Inbound Unknown Protocols       = 5813
Outbound Octets                 = 1673233
Outbound Unicast Packets        = 20981
Outbound Non-Unicast Packets    = 12
Outbound Discards               = 0
Outbound Errors                 = 0
Outbound Queue Length           = 0
Specific                        = 655367

Ethernet-like Statistics Group
Index                           = 0
Alignment Errors                = 0
FCS Errors                      = 0
Single Collision Frames         = 0
Multiple Collision Frames       = 0
Deferred Transmissions          = 0
Late Collisions                 = 0
Excessive Collisions            = 0
Internal MAC Transmit Errors    = 0
Carrier Sense Errors            = 0
Frames Too Long                 = 0
Internal MAC Receive Errors     = 0
```

The output of **lanadmin** is tremendous. Detailed knowledge about the data link layer protocols is necessary to understand all of the information offered by **lanadmin**. The following are only a few tips on how to use and interpret the information that **lanadmin** displays:

© 2003 Hewlett-Packard Development Company, L.P.

PPA -Physical Point of Attachment	The Physical Point of Attachment (PPA) number of the LAN interface.
Type (value)	LAN interface type. (IEEE 802.3/Ethernet interface in the preceding example.)
MTU Size	The maximum transfer unit is the maximum size of a frame. The default for Ethernet and IEEE 802.3 interfaces is 1,500 bytes.
Speed	Maximum transfer rate of the interface. (10 Mbps in the example.)
Station Address	Link level address (MAC level address).
Administration Status	Up means that the autoconfiguration of the LANIC was successfully completed. Down means that the LANIC is defective or no kernel driver for this interface is configured.
Operation Status	Up means the LANIC was successfully powered up by the **ifconfig** command.

To interpret all other values, look for lines with terms like *Discards*, *Errors*, *Collision*, *Deferred*, and *Too Long*.

Lines with values that are not equal to 0 are not necessarily a problem. If you have a real problem in OSI layer 1 or 2, **lanadmin** will show some lines with very high values.

Produce an output listing of **lanadmin** when you do not have any problems with your network and keep this listing. Compare this listing with the **lanadmin** output you get when problems occur. This information is very helpful when troubleshooting your network.

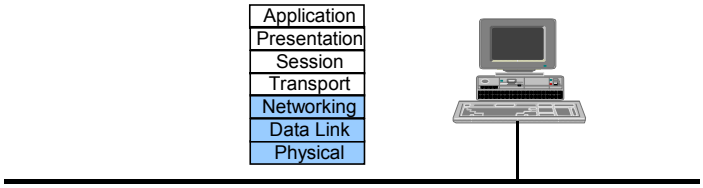
To produce **lanadmin** output with a shell script, do the following:

```
lanadmin -te > listing.lanadmin <<!
lan
display

quit
!
```


© 2003 Hewlett-Packard Development Company, L.P.

6–7. SLIDE: The `arp` Command

The `arp` Command



- ARP is the address resolution protocol.
- The arp command is used to display and modify entries in the ARP table.
- Options which modify the ARP table require root privilege.

Example

```
# /usr/sbin/arp -a
frank (192.6.30.1) at 0:60:b0:7:4c:4d ether
beverly (192.6.30.5) at 0:60:b0:7:c1:79 ether
jeff (192.6.30.4) at 0:60:b0:7:e1:12 ether
bill (192.6.30.2) at 0:60:b0:7:7e:69 ether
larry (192.6.30.3) at 0:60:b0:7:e1:a2 ether
```

Student Notes

The **`/usr/sbin/arp`** command displays or modifies the entries in the ARP kernel table that relate Internet (level 3) to Ethernet (level 2) addresses used by the ARP protocol. It has several options, some of which can only be used by a superuser.

Syntax:

`arp` ***`hostname`***	Displays the current ARP entry for ***`hostname`***.
`arp -a [`*`system`*`][`*`core`*`]`	Displays all current ARP entries by reading the table from file ***`core`*** (default **`/dev/kmem`**) based on the kernel file ***`system`*** (default **`/stand/vmunix`**).
`arp -d` ***`hostname`***	If an ARP entry exists for the host called ***`hostname`***, then delete it. This requires superuser privileges.
`arp -s [`*`parameter`*`]`	Create an ARP entry for a host with a new Ethernet address. This requires superuser privileges.

© 2003 Hewlett-Packard Development Company, L.P.

arp -f *filename*	Read file ***filename*** and set multiple entries in the ARP tables. Entries in the file should be of the form ***hostname address*** **[temp] [pub] [trail]**. This requires superuser privileges.

If a defective LAN interface is replaced by a new one, remember that the new unit will have a new link level address. Any remote host that has still the old link level address in its ARP table will not be able to communicate with this replacement interface. You must delete the wrong entry from the ARP tables on these remote hosts.

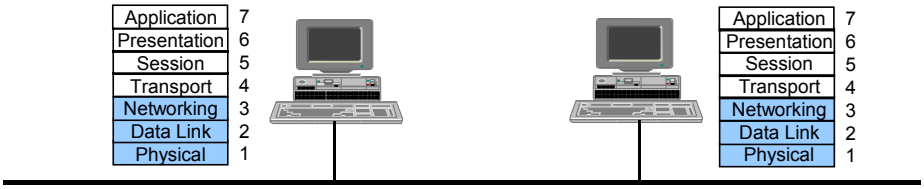
If you want to know the link level address of a remote host in your network, you can send a **ping** to this host and read then your ARP table.

For more information, see the **man** pages for **arp(1M)** and **arp(7)**.

© 2003 Hewlett-Packard Development Company, L.P.

6–8. SLIDE: The `ping` Command

The `ping` Command



- The `ping` command tests the IP connectivity to a remote system.

Example

```
# ping bill
PING 192.6.30.2: 64 byte packets
64 bytes from 192.6.30.2: icmp_seq=0. time=43. ms
64 bytes from 192.6.30.2: icmp_seq=1. time=223. ms
64 bytes from 192.6.30.2: icmp_seq=2. time=199. ms
64 bytes from 192.6.30.2: icmp_seq=3. time=170. ms

----bill PING Statistics----
5 packets transmitted, 4 packets received, 20% packet loss
round-trip (ms)  min/avg/max = 43/158/223
```

Student Notes

`ping` tests up through the network layer (layer 3) of the OSI model. Any user can execute **`ping`**.

When you encounter a network problem, it is typically a good idea to execute the **`ping`** command first. If **`ping`** is successful in transferring packets, you can typically rule out problems below layer 3 (hardware problems such as bad cables or transceivers), and you can run tests on the upper layers. If **`ping`** fails, you should use **`lanadmin`** or **`lanscan`** to diagnose your LAN hardware.

Use **`ping`**

- to do a preliminary connectivity check when setting up new nodes.
- when difficulties arise in connecting to a particular node.

© 2003 Hewlett-Packard Development Company, L.P.

Syntax

```
ping hostname [packet_size] [-n [num_packets]]
```

in which

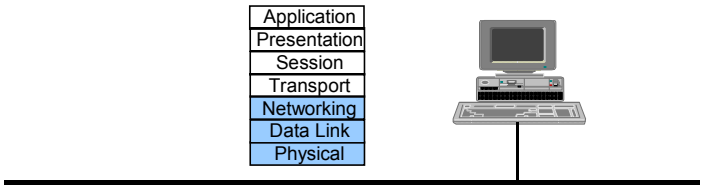
hostname	The IP address or the official host name.
packet_size	By default, the size of transmitted packets is 64 bytes. The minimum value for packet size is eight bytes and the maximum is 4,096 bytes. If *packet_size* is less than 16 bytes, there is not enough room for timing information, so round-trip times will not be displayed.
num_packets	The number of packets **ping** will transmit before terminating. By default, **ping** will send packets until interrupted by pressing CTRL+c. If you do not specify a packet size, you need to use **-n** *num_packets*.

NOTE: If you use **ping** on your local host (loopback), you test just the network layer (layer 3). The test could be successful even if the LAN hardware is down.

© 2003 Hewlett-Packard Development Company, L.P.

6–9. SLIDE: The `netstat -i` Command

The `netstat -i` Command



- The `netstat -i` command displays a LAN interface status report.
- The `netstat -in` command displays IPs instead of hostnames.
- An asterisk (`*`) in the output indicates the interface is down.
- The output of `netstat -i` varies from HP-UX 10.x to HP-UX 11.00.

Example for HP-UX 11.00

```
# netstat -i
Name            Mtu Network          Address          Ipkts      Opkts
lo0            4136 127.0.0.0        localhost          838        838
lan0           1500 192.6.30.0       bill            160952     111715
```

Student Notes

The **`netstat`** command reports network and protocol statistics regarding traffic and the status of the local LAN interface. Any user can execute **`netstat`**.

There are many options to **`netstat`**. The most useful options are those that display information that is not available through other commands (such as **`ping`**, **`lanscan`**, and **`lanadmin`**). Within this module, we will discuss only the following options, which display information about OSI layers 1, 2, and 3:

`-n` Used in conjunction with other options, this option shows IP network addresses as numbers in dot notation (instead of names).

`-i` Shows the state of the network interfaces. This includes both primary and logical interfaces.

`-r` Lists all routes in the local routing tables. When **`-v`** is used with the **`-r`** option, **`netstat`** also displays the network masks in the route entries. The **`-r -s`** combination is not supported in HP-UX 11.0.

`-s` Displays routing statistics.

© 2003 Hewlett-Packard Development Company, L.P.

The **netstat -i** command shows information about the status of all LAN interfaces as well as a table of cumulative statistics regarding packets transferred. In version 10.20 and earlier, there was information on collisions and errors as well. The cumulative statistic starts with powering up the interface. It can be reset by the **reset** functionality of the **lanadmin** command.

Name — Name of the network interface.

- **lan0** is your first IEEE 802.3/Ethernet network interface.
- **lan1** is your second network interface. The hardware path is displayed by **lanscan**.
- **lo0** refers to your local loopback interface (IP address 127.0.0.1).
- **ni0** and **ni1** are two built-in RS 232 interfaces. They are possible network interfaces. You can configure them with the serial line interface protocol (SLIP) to use the IP protocol in a point-to-point serial network. For more information, see the **man** page **pppd(1)**.

  The asterisk (*) shows that the interface was not activated.

Mtu — Maximum transmission unit shows the biggest possible size of a frame. With IEEE 802.3 it is 1500 Bytes.

Network — Shows the IP address or the name of the network to which this interface belongs. If there is a name, the file **/etc/networks** is configured. **none** indicates that the interface is not powered up.

Address — Shows the IP address or the name of the interface. If there is a name, the IP address was translated by the **hosts** file, NIS, or **BIND**. **none** indicates that the interface is not powered up.

Ipkts — Number of input packets received.

Opkts — Number of output packets transmitted.

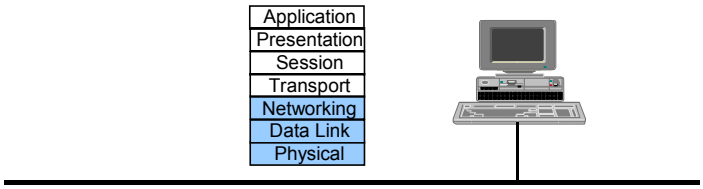
To determine the number of packets going over the network, use the **netstat *interval*** option. Network traffic through the local network interface will be reported every ***interval*** seconds. The first line and every 24th line thereafter show cumulative statistics since the system was powered up or the statistics were reset with **lanadmin**. The slide shows the number of packets transmitted and received, the number of packets with errors, and the number of collisions.

Most of this information can also be gathered with **lanadmin**. The difference is that **lanadmin** provides a snapshot view (a single sample), whereas **netstat** is continuously sampling.

© 2003 Hewlett-Packard Development Company, L.P.

6–10. SLIDE: The `netstat -r` Command

The `netstat -r` Command



- The `netstat -r` command displays all routes defined in the route table.
- The `netstat -rn` command displays IP addresses instead of hostnames.

Example

```
# netstat -rn
Routing tables
Dest/Netmask            Gateway             Flags  Refs   Interface  Pmtu
127.0.0.1               127.0.0.1           UH        0   lo0         4136
192.6.30.2              192.6.30.2          UH        0   lan0        4136
192.6.30.0              192.6.30.2          U         2   lan0        1500
127.0.0.0               127.0.0.1           U         0   lo0         4136
default                 192.6.30.1          UG        0   lan0        1500
```

Student Notes

`netstat -r` shows your host's routing tables. By default, **`netstat`** resolves IP addresses to hostnames. If you wish to view IP addresses in the routing table, use the **`-n`** option in addition to **`-r`**.

- The Dest/Netmask field identifies the destination host or network for each table entry.
- The Gateway field identifies the next hop required to get to each of the destinations.
- The Flags field may contain any or all of U, G, or H.

  U The router is up and running.

  G The router entry is a gateway (means a remote router).

  H The destination is a host, not a network.
- The Refs field gives the current number of active uses of the route.
- Pmtu is the maximum transmission unit (maximum frame size).

© 2003 Hewlett-Packard Development Company, L.P.

If you have only one LAN interface, you should have a minimum of four entries in your routing table:

- A route to the loopback address (127.0.0.1)
- A route to the loopback network (127.0.0.0)
- A route to your own IP address through your own interface card.
- A route to your own IP network through your own interface card.

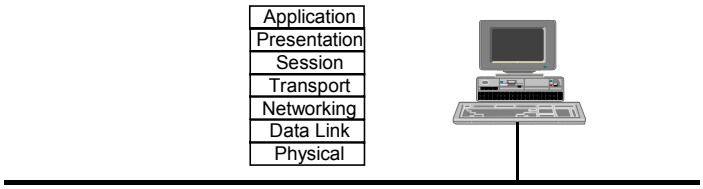
Each time you configure an additional logical interface via the **`ifconfig`** command, HP-UX automatically adds that IP address to your routing table, as well as a route to the network to which your new interface is attached.

Entries can be added to and removed manually from the routing table via the **`route`** command.

© 2003 Hewlett-Packard Development Company, L.P.

6–11. SLIDE: The `nslookup` Command

The `nslookup` Command



- The `nslookup` command resolves hostnames to IP addresses.
- The `nslookup` command is useful for identifying problems with `/etc/hosts`.
- The `nslookup` command contains other capabilities which operate at different layers of the OSI model.

Example

```
# nslookup mickie
Using /etc/hosts on:  bill

Name:    mickie
Address:  192.6.30.3
```

Student Notes

The **`nslookup`** command checks how the local system resolves host names to IP addresses:

```
$ nslookup
Default Name Server: chris.hp.com
Address: 192.6.21.2

> Ctrl+d

$ nslookup darren
Default Name Server: chris.hp.com
Address: 192.6.21.2

Name: darren.hp.com
Address: 192.6.21.4
```


© 2003 Hewlett-Packard Development Company, L.P.

Some other useful **nslookup** built-in commands are:

> *host server*	Looks up information for host using name server
>ls -d *domain*	Lists all information for domain (can be long...)
>ls -d *domain* > file	Lists all information for domain and redirect it to file
>set debug	Turns debugging mode on
>set all	Prints the current values of the various options that have been set
>policy	Prints the order of precedence in the IP address lookup sequence.

© 2003 Hewlett-Packard Development Company, L.P.

6–12. LAB: Troubleshooting Network Connectivity

Directions

Answer all questions below. Also, record the commands you use to find the answers.

Preliminary

Over the course of this lab, you will be asked to disable your LAN card, which can cause serious problems for CDE. Before starting the lab, shut down CDE:

```
# /sbin/init.d/dtlogin.rc stop
```

Part 1: Determining Your Current Network Configuration

1. Determine your host name, and MAC address and IP address of your LAN interface(s).

```
MAC address(es) :
IP address(es)  :
Hostname        :
```

2. To which network are you directly connected?

   Do you have a default route defined so you can reach other networks?

3. Given a host name, how can you determine the IP address of another machine? Ask your neighbor for their host name, and then determine their IP.

4. Now that you know your neighbor's IP address, how can you determine their MAC address? Do it.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Testing LAN Connectivity

1. Ensure that your lan0 card and your neighbor's lan0 interface card are both in an "UP" state. Can you ping your neighbor's IP address?

2. What happens if your LAN card is "DOWN"? Change the IP configuration state of your lan0 interface to "DOWN." What appears in **`netstat -i`** now for your card?

3. While your LAN card is DOWN, can you ...
   - Ping your neighbor's IP address?
   - Ping your own IP address?
   - Ping your loopback address?

4. Now try the **`linkloop`** command on your neighbor's MAC address. Does this work? Explain.

5. Based on your answer to the previous question, when might **`linkloop`** be useful?

6. Bring your lan0 card back to an "UP" state.

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Troubleshooting Connectivity Problems

1. Before starting this exercise, make sure you are able to **`ping`** host name "**`corp`**," at IP address 128.1.0.1.

2. There should be a shell script in your **`/labs`** directory called **`/labs/corrupt.sh`**. Run the script. When prompted, enter a number between 1 and 6. Based on your response, the script will corrupt your LAN configuration in one of six different ways. When the script terminates, your task is to fix your LAN configuration so the command **`ping corp`** succeeds. Take advantage of all the tools we discussed in this chapter.

3. Once you successfully troubleshoot and fix your configuration, run the script again, choose a different number, and again fix the resulting problem. If time permits, try each of the 6 options provided by the **`corrupt.sh`** script.

Good luck!

Part 4: Cleanup

1. Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

Module 7 — Starting Network Services

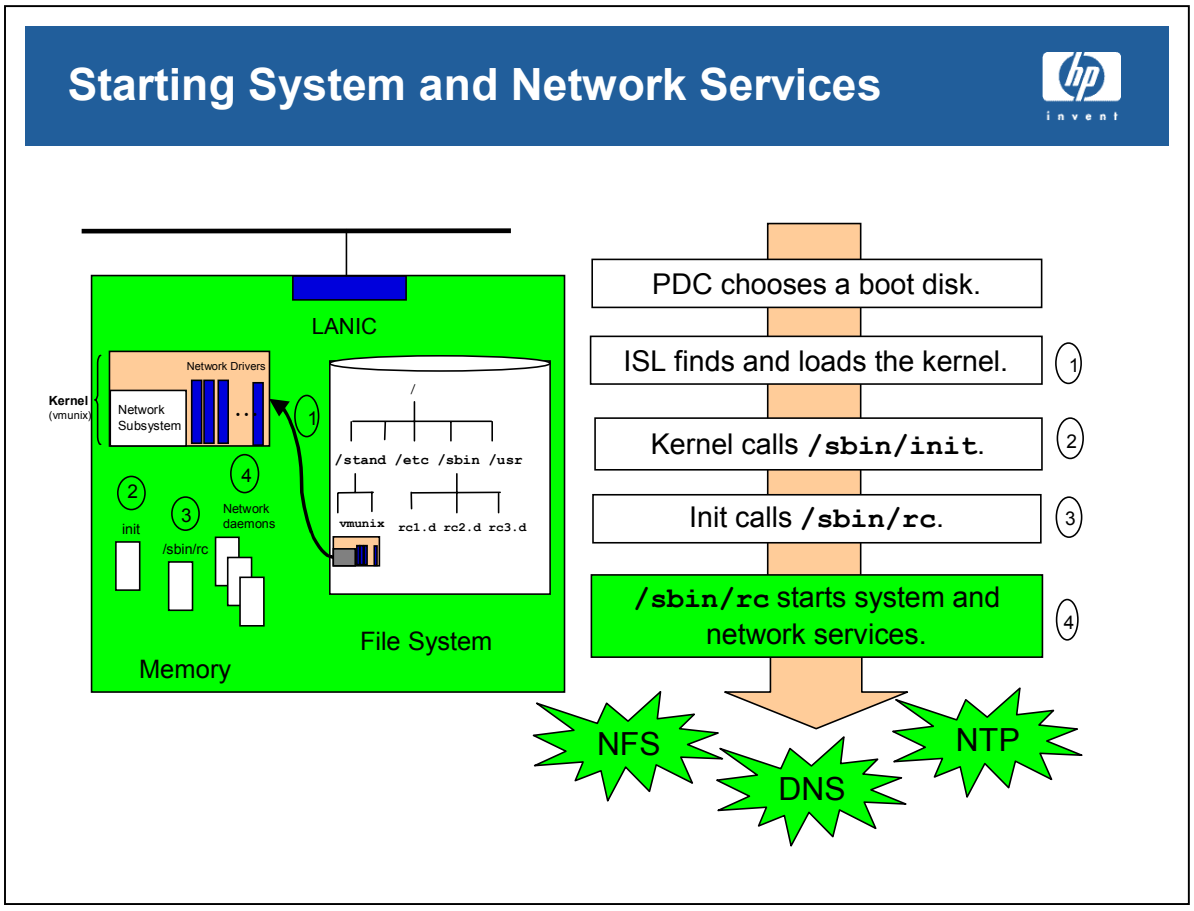
Objectives

Upon completion of this module, you will be able to do the following:

- Describe how run levels are used during system boot time.
- Change and view the system's current run level.
- Define the default system run level.
- Enable and disable services using the **`/etc/rc.config.d config`** files.
- Create custom startup and shutdown scripts to start additional services during the boot process.
- View the startup error log file.

© 2003 Hewlett-Packard Development Company, L.P.

7–1. SLIDE: Starting System and Network Services



Student Notes

In earlier chapters, we walked through the process of configuring a LAN interface and connecting an HP-UX system to a network. After configuring a LAN interface, numerous services can be configured to use the system's LAN connection. The slide above lists just a few examples:

- **NFS**: Makes it possible to access file systems across the network.
- **DNS**: Is a network service that resolves host names to IP addresses.
- **NTP**: Can be used to synchronize the system clocks on the LAN.

These services, as well as many other system services such as **`cron`** and **`lp`** require a daemon to be running on the system. This chapter will discuss the process used by HP-UX to start these daemons during a system boot, and kill them during system shutdown.

Review of the Early Steps in the System Boot Process

The early stage of the system boot process simply finds and loads the kernel into memory. Immediately after the system is powered on, the "Processor Dependent Code" (PDC) is

© 2003 Hewlett-Packard Development Company, L.P.

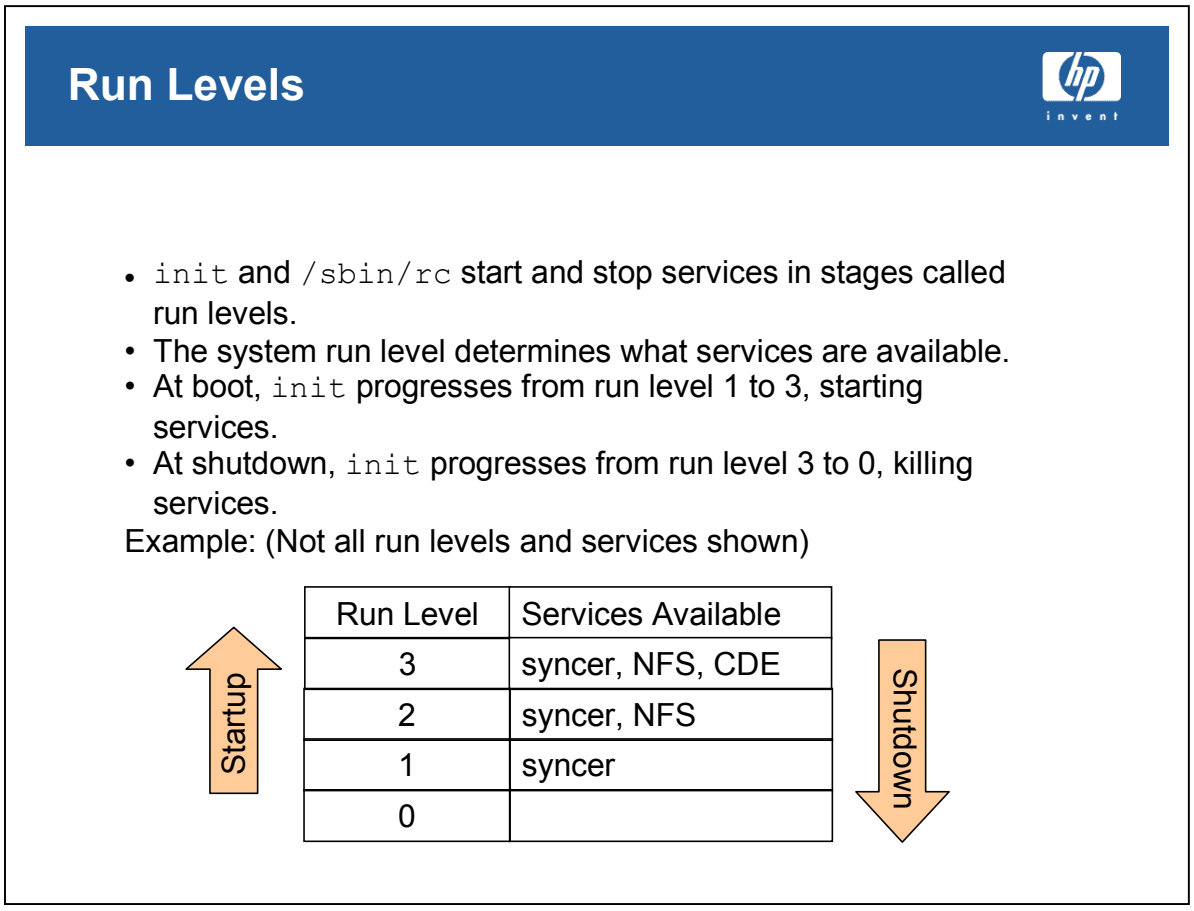
loaded in memory from the system's BootROM chip. The PDC does an initial hardware test, then checks stable storage to determine which disk is the default boot disk.

Each boot disk contains a boot area that includes an "Initial System Loader" executable. The ISL calls the HP-UX kernel loader, which then loads the kernel in memory. The kernel does a sanity check on the root file system, and then calls the **`init`** daemon. The **`init`** daemon is responsible for bringing the system to a fully functional state. The **`init`** daemon performs some of the system initialization tasks itself. It checks for corruption in the file systems listed in **`/etc/fstab`**, initializes the system console, and performs several other tasks defined in **`/etc/inittab`**.

`init` calls on the **`/sbin/rc`** program, however, to start most of the system services such as NFS, DNS, and NTP that are required to bring the system to a fully functional state.

© 2003 Hewlett-Packard Development Company, L.P.

7–2. SLIDE: Run Levels



Student Notes

Numerous services must be started to bring an HP-UX system up to a fully functional state. There may be some dependencies to consider as all of these services are starting. For example, it would not make sense to start Networked File System functionality until the LAN cards have been configured. So how does **`init`** guarantee that these dependencies are met?

Introduction to Run-Levels

The **`init`** daemon brings the system up to a fully functional state in stages known as "run levels." A run level is a system state in which a specific set of processes is allowed to run. The run level your system is at determines what functionality and services are available.

- More services are available at higher run levels.
- Fewer services are available at lower run levels.

© 2003 Hewlett-Packard Development Company, L.P.

Valid run levels in HP-UX range include **0**, **s**, **S**, **1-6**:

Run-level **0**	Reserved for system shutdown. When running in run-level 0, the system performs the normal shutdown procedure, thereby stopping all processes and halting the system.
Run-level **s**	is a special run-level reserved for system administration tasks. It is also referred to as single-user run-level meaning it is reserved for a single user, typically, the system administrator. For example, shutting down the system (**/sbin/shutdown**) brings you to run-level s.
Run-level **S**	Similar to run-level s. In run-level s, only the physical system console has access to the operating system, whereas in run-level S the capabilities of the system console are switched to the terminal where you are logged in, thus making it the virtual system console.
Run-level **1**	Similar to single-user, but file systems are mounted and the syncer is running. This run level can also be used to perform system administrative tasks.
Run-level **2**	Multiuser state. This run level allows all users to access the system.
Run-level **3**	For HP CDE users, HP CDE is active at this run level. Beginning with HP-UX release 10.20, CDE is the default user desktop environment. Also, at run-level 3, NFS file systems are exported; this capability is called Networked Multiuser state.
Run-level **4**	For HP VUE users. In this mode, HP VUE is active, providing the operating system release is 10.30 or below. As of HP-UX 11.00, HP VUE is no longer supported.

Run-Levels and the Startup/Shutdown Procedure

Initially, **init** brings the system to run-level **1**, then **2**, then **3**, and so forth until it reaches the default run level defined by the **init** default line in **/etc/inittab**. At each run level, **init** calls **/sbin/rc** to start additional services.

At system shutdown, then, **init** brings the system down to run-level 0 one run-level at a time. At each run-level, **/sbin/rc** has an opportunity to kill whatever services are no longer needed.

Changing and Viewing the System Run-Level

You can determine your current run level with the **who -r** command. You may also change your system run level with the **init** command:

```
# who -r      # check your current run-level
# init 4      # move up to run-level 4
# init 2      # move down to run-level 2
# init 3      # move back up to run-level 3
```


© 2003 Hewlett-Packard Development Company, L.P.

Questions

1. Try the **`init`** command to change run-levels a few times. What happened when you moved up to run-level **`4`**? Did any additional services appear to start?

2. What happened when you moved from run-level **`4`** to run-level **`2`**? Did any services disappear?

3. How might changing run levels affect your users?

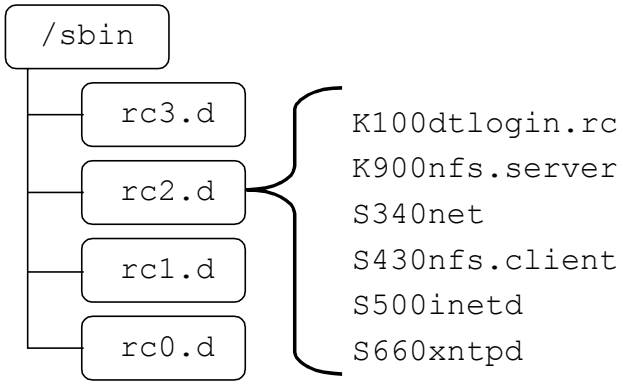
4. When might it be useful to change run levels?

© 2003 Hewlett-Packard Development Company, L.P.

7–3. SLIDE: `/sbin/rc*.d` Directories

/sbin/rc*.d Directories

- `/sbin/rc*.d` directories determine at which run levels services start and stop.
- `/sbin/rc` runs S scripts to start services during system startup.
- `/sbin/rc` runs K scripts to kill services during system shutdown.



Student Notes

At each run level, the **`init`** daemon calls **`/sbin/rc`** to start any necessary system and network services. The **`/sbin/rc`** program determines which services to start and stop at the new run level by consulting one of the **`/sbin/rc*.d`** directories.

There is one **`/sbin/rc*.d`** directory for each defined system run level:

```
/sbin/rc0.d
/sbin/rc1.d
/sbin/rc2.d
/sbin/rc3.d
```

The **`/sbin/rc*.d`** directories contain "S" and "K " scripts. " S" scripts start services, while "K" scripts stop (**`kill`**) services. Most services started by **`/sbin/rc`** have both an "S" script and a "K" script in the **`/sbin/rc*.d`** directories. You can use the **`ls`** command to see which services are started at each run level:

```
# ls /sbin/rc*.d/*
```


© 2003 Hewlett-Packard Development Company, L.P.

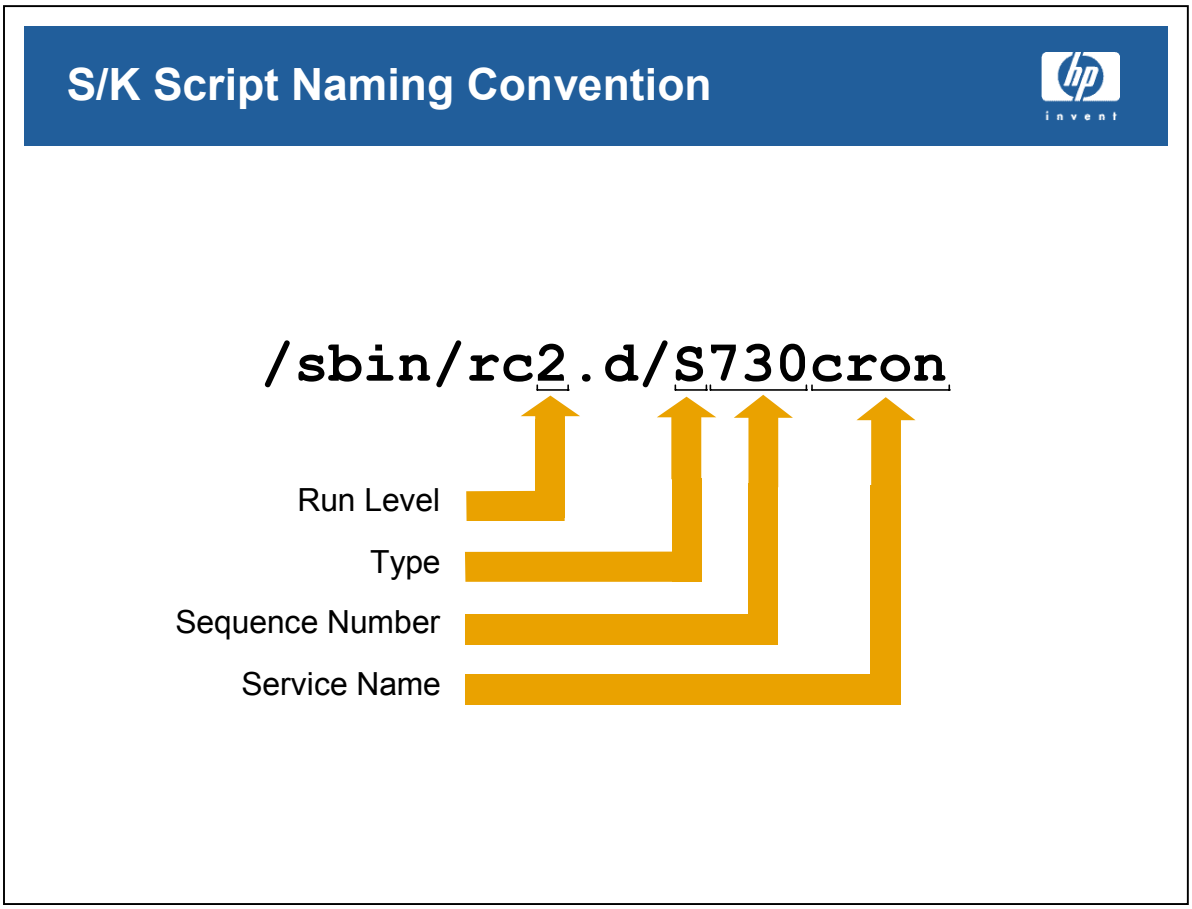
Questions

1. Do an **`ls /sbin/rc*.d/*`**. At which run level are the majority of the system services and daemons started? Which **`rc*.d`** directory contains the most kill scripts?

2. If a service's "S" script is in **`/sbin/rc2.d`**, where would you expect to find its "K" script? Do an **`ls /sbin/rc*.d/*`** to see if your hypothesis is true.

© 2003 Hewlett-Packard Development Company, L.P.

7–4. SLIDE: S/K Script Naming Convention



Student Notes

There are several components to each S/K script name.

The first character in each script name simply indicates whether the script should be called to start a service (S) or kill a service (K).

The second component of each script name is a "sequence number." When **`init`** brings the system to a higher run-level, **`/sbin/rc`** executes the "S" scripts in the appropriate **`/sbin/rc*.d`** directory in ascending order by sequence number. When **`init`** brings the system to a lower run-level, **`/sbin/rc`** executes the "K" scripts in the appropriate **`/sbin/rc*.d`** directory in ascending order by sequence number. This allows **`/sbin/rc`** to accommodate dependencies within a run level.

The final component of each script name simply identifies the service or daemon with which the S/K script is associated.

Assigning Sequence Numbers

In order to meet dependency requirements, services are generally killed in the reverse order from which they are started.

© 2003 Hewlett-Packard Development Company, L.P.

For example, assume there are four services, W, X, Y, and Z. The S/K script names for these services would likely be:

```
/sbin/rc3.d:      /sbin/rc2.d:
------------      ------------
S200W             K800W
S300X             K700X
S400Y             K600Y
S500Z             K500Z
```

What appears to be the relationship between start and kill sequence numbers?

NOTE: S/K sequence numbers may range in value from 100 to 900.For custom S/K startup scripts that you create, HP recommends that you use the generic start and kill sequence numbers:

Generic start sequence number: 900
Generic kill sequence number: 100

Questions

Consider the following sample S/K scripts and answer the questions that follow:

```
/sbin/rc2.d/K900nfs.server
/sbin/rc2.d/S340net
/sbin/rc2.d/S430nfs.client
/sbin/rc2.d/S500inetd
/sbin/rc2.d/S660xntpd
```

1. When moving up to run-level 2, which services would be started, and in which order?

2. When moving down to run-level 2 from run-level 3, which services would be stopped, and in which order?

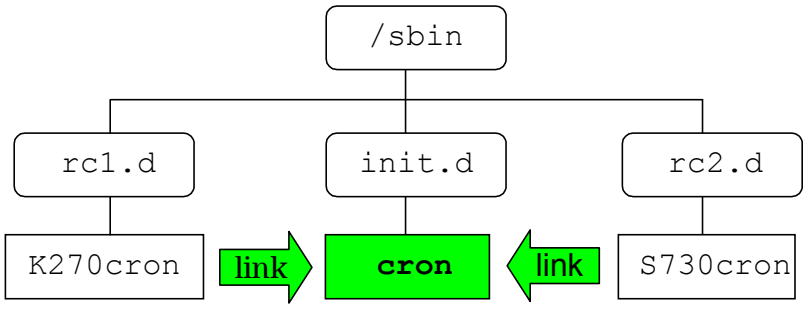
3. Write the full path names for the "K" scripts that you would expect to be associated with each of the "S" scripts shown above.

4. Write the full pathname of the **`S`** script that would correspond to the **`nfs.server`** kill script shown above.

© 2003 Hewlett-Packard Development Company, L.P.

7–5. SLIDE: `/sbin/init.d/*` Scripts

/sbin/init.d/* Scripts



- Every service started by `/sbin/rc` has an associated script in `/sbin/init.d`.
- `/sbin/init.d` scripts contain code needed to actually start/kill services.
- `/sbin/rc*.d/*` scripts are just symbolic links to `/sbin/init.d` scripts!

Student Notes

If you do a long listing of the **`/sbin/rc*.d`** directories, you will note that the S/K scripts aren't really scripts at all.

Each service started by **`/sbin/rc`** has a shell script in the **`/sbin/init.d`** directory. These scripts contain the commands necessary to both start AND stop their associated services. The files in the **`/sbin/rc*.d`** directories are actually nothing more than symbolic links to scripts in the **`/sbin/init.d`** directory.

© 2003 Hewlett-Packard Development Company, L.P.

7–6. SLIDE: What's in an `init.d` Script?

What's in an `init.d` Script?

- Scripts in `/sbin/init.d` accept a single argument.
- Scripts do one of four things, depending on the argument value.

Sample `init.d` script (simplified):

`/sbin/init.d/cron:`

```
case $1 in
   start_msg) echo "Start clock daemon"
   stop_msg)  echo "Stop clock daemon"
   start)     # Commands to start cron
   stop)      # Commands to kill cron
esac
```

Student Notes

All of the scripts in the **`/sbin/init.d`** directory have essentially the same structure. All are built around a case statement that evaluates the first argument passed to the script (**`$1`**). The scripts recognize four valid values for this first argument:

`start_msg` — The **`start_msg`** argument simply echoes a message indicating what service or daemon the script controls. **`/sbin/rc`** uses the **`start_msg`** argument to generate the checklist of services that appears on the system console during system startup.

`stop_msg` — The **`stop_msg`** has much the same purpose as the **`start_msg`** argument. **`/sbin/rc`** calls the **`/sbin/init.d`** scripts with **`stop_msg`** to generate the shutdown checklist that appears on the console during system shutdown.

`start` — When called with the start argument, the **`/sbin/init.d`** scripts execute whatever commands are necessary to actually start the associated service.

© 2003 Hewlett-Packard Development Company, L.P.

stop	When called with the stop argument, the **/sbin/init.d** scripts execute whatever commands are necessary to actually stop the associated service.

Starting and Stopping Services Manually

Usually, **/sbin/rc** calls the **/sbin/init.d** scripts automatically during startup and shutdown. However, you can also manually start or stop a service. The example below might be used to manually start or stop the cron daemon:

```
# /sbin/init.d/cron start
# /sbin/init.d/cron stop
```


© 2003 Hewlett-Packard Development Company, L.P.

7–7. SLIDE: `/etc/rc.config.d/*` Files

/etc/rc.config.d/* Files

- You may wish to disable a service that's not needed, or enable a new service.
- Services may be enabled or disabled via control variables.
- Control variables are defined in files under `/etc/rc.config.d`.
- `/sbin/init.d/` scripts source `/etc/rc.config.d/*` files to determine control variable values.

`/etc/rc.config.d/cron`

```
CRON=1 # Set control variable to 1 to enable
       # Set control variable to 0 to disable
```

`/sbin/init.d/cron (simplified)`

```
case $1 in
   start_msg) echo "Start clock daemon"
   stop_msg)  echo "Stop clock daemon"
   start)     if CRON=1 then start the cron daemon
   stop)      if CRON=1 then kill the cron daemon
esac
```

Student Notes

In addition to an **`/sbin/init.d`** script, most services also have an associated configuration file in the **`/etc/rc.config.d`** directory. These configuration files allow the administrator to:

- Disable unneeded daemons/service
- Change parameters to customize a service's behavior

Enabling/Disabling Services with Control Variables

Most **`init.d`** scripts check a control variable to determine if the associated service should be started.

- Control variable = 1 --> Script should run at startup/shutdown.
- Control variable = 0 --> Script should not run at startup/shutdown.

The control variable usually takes the name of the service it controls.

© 2003 Hewlett-Packard Development Company, L.P.

- Control variable for **/sbin/init.d/cron**: CRON.
- Control variable for **/sbin/init.d/nfs.server**: NFS_SERVER.
- Control variable for **/sbin/init.d/nfs.client**: NFS_CLIENT.

The values of these control variables are set in the configuration files under the **/etc/rc.config.d** directory. Some **/sbin/init.d** scripts have their own, dedicated configuration files in **/etc/rc.config.d**, but some services share a common configuration file.

Examples

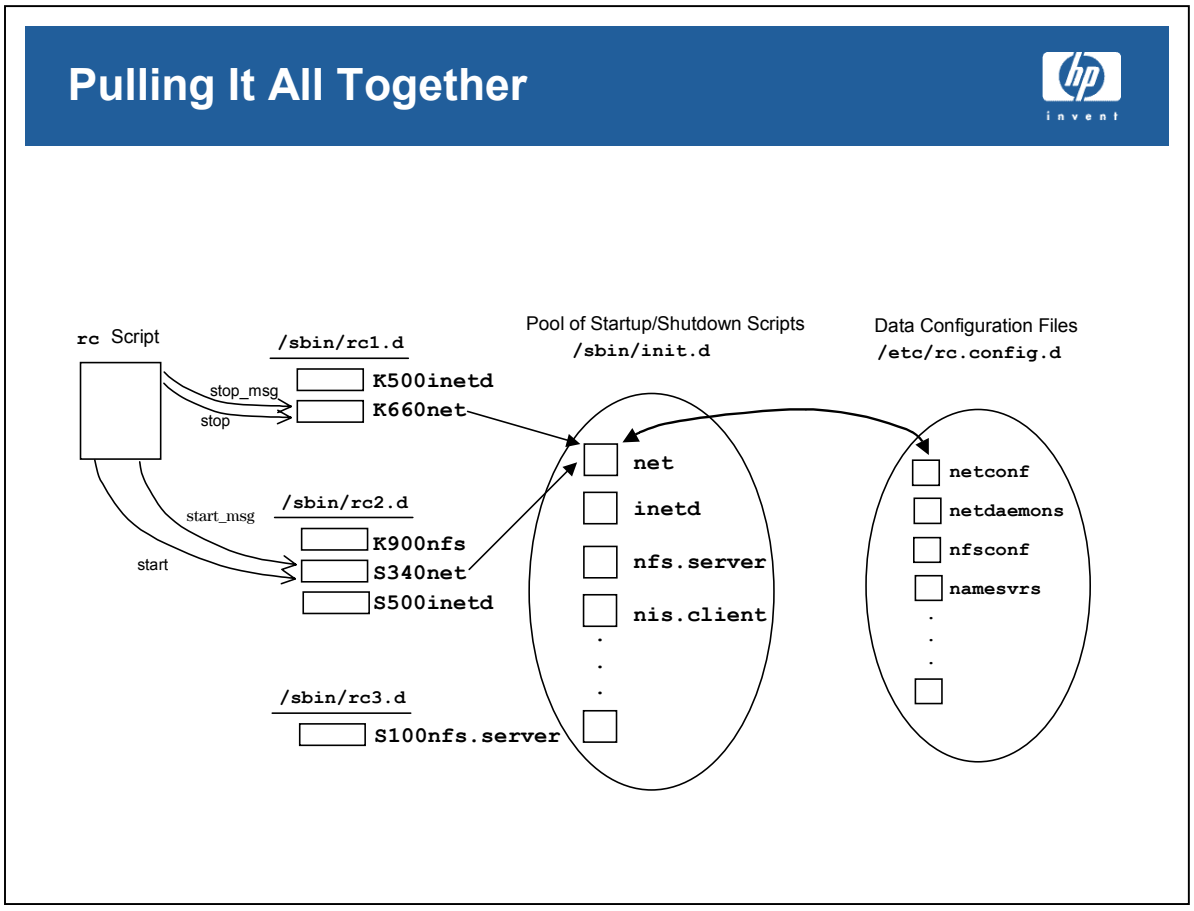
```
/sbin/init.d script   /etc/rc.config.d file      control variable
-------------------   ---------------------      ----------------
cron                  /etc/rc.config.d/cron      CRON
nfs.client            /etc/rc.config.d/nfsconf   NFS_CLIENT
nfs.server            /etc/rc.config.d/nfsconf   NFS_SERVER
```

Many configuration files set other parameters used by the startup script, too. Recall that the **/etc/rc.config.d/netconf** file, for example, defined the system host name, IP address, and routing information.

WARNING:	Never modify the scripts in **/sbin/init.d** directly. Modify startup script parameters via the **/etc/rc.config.d** config files.

© 2003 Hewlett-Packard Development Company, L.P.

7–8. SLIDE: Pulling It All Together



Student Notes

The above slide summarizes all the files and directories involved in starting and shutting down processes/daemons at startup and shutdown, and shows how the files and directories interact.

The graphics recap the concepts presented on the five previous slides, including:

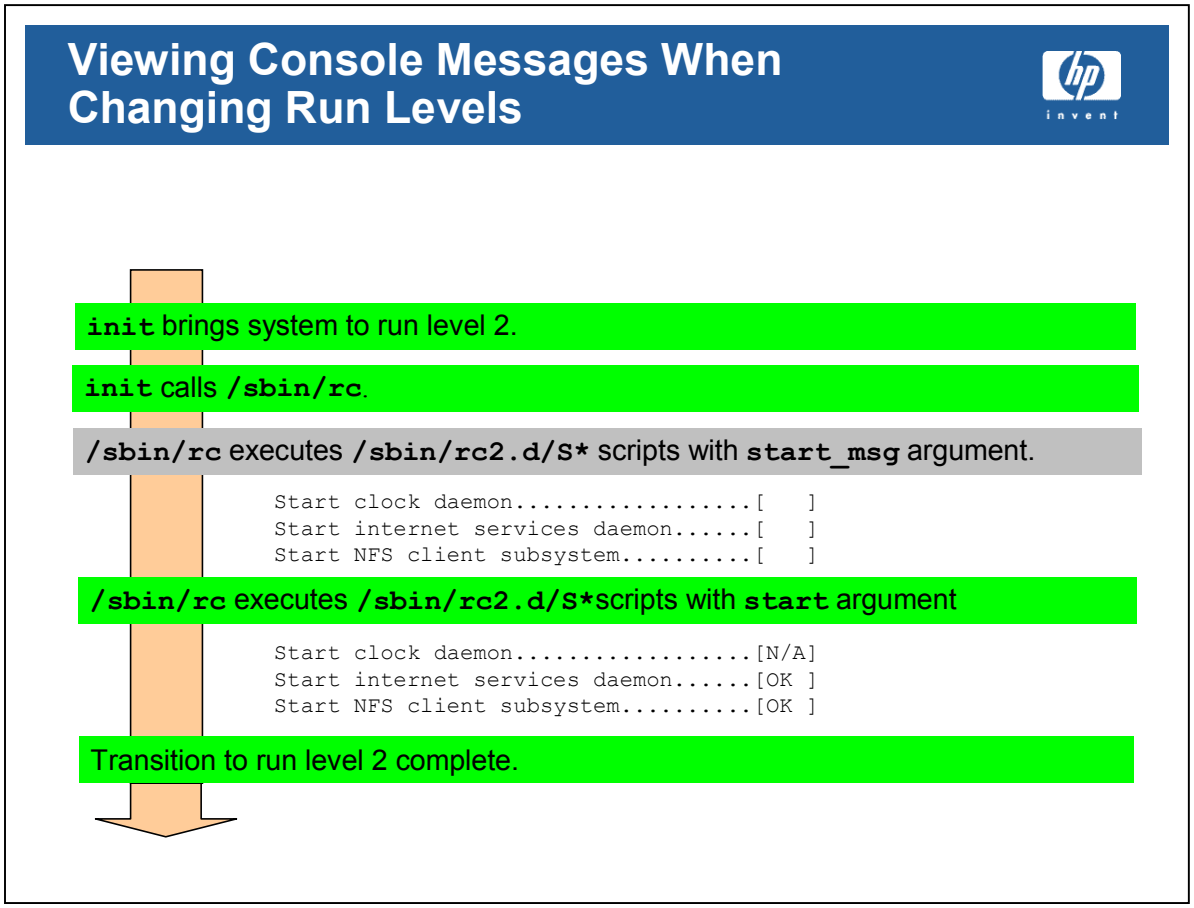
The **`/sbin/rc*.d`** directories	These directories, also known as run level directories, contain the names of scripts to execute when transitioning to the various run levels.
The S/K naming convention	Within the **`/sbin/rc*.d`** directories (run-level directories), all scripts followed a pre-defined naming convention which indicated whether to Start or Kill a daemon, and the order in which the scripts were to execute.
The **`/sbin/init.d`** directory	This directory contained all the executable scripts. These scripts are referenced via symbolic links from the **`/sbin/rc*.d`** run level directories.

© 2003 Hewlett-Packard Development Company, L.P.

The contents of the **`init.d`** scripts	Each executable script contained instructions for starting and stopping the processes/daemons associated with the subsystem.
The **`/etc/rc.config.d`** directory	This directory contained customization files for all the executable scripts in **`/sbin/init.d`**. Because the executables should NOT be modified directly, the customization for these scripts were kept in separate files located under this directory.

© 2003 Hewlett-Packard Development Company, L.P.

7–9. SLIDE: Viewing Console Messages When Changing Run Levels



Student Notes

During the transition from one run-level to another, a checklist of all the actions to be performed during the transition will appear on the screen. The **/sbin/rc** program creates the checklist by calling each execution script with an argument of **start_msg** (if transitioning to a higher run level) or **stop_msg** (if transitioning to a lower run level).

Once the checklist is created, the **/sbin/rc** program calls each execution script again, this time with an argument of **start** or **stop**. This invocation attempts to either start or stop the subsystem. The outcomes of this second invocation is indicated on the checklist screen (the far right side) with one of the following status:

OK The execution script successfully started up (or shutdown) the subsystem.

FAIL The execution script was unable to start (or stop) the subsystem. When an execution script fails, a message will appear at the bottom of the screen, stating:

```
* - An error has occurred!
```


© 2003 Hewlett-Packard Development Company, L.P.

```
* - Refer to the file /etc/rc.log for more information
```

N/A The execution script did not try to start (or stop) the subsystem because it was disabled in the **/etc/rc.config.d** configuration file.

When Things Go Wrong ...

Occasionally, a misconfigured **/etc/rc.config.d/** file, or some other problem on the system may cause startup scripts to hang or fail. In most cases, you can terminate the currently running startup script and escape to a console login by hitting **Control-**. Check the **/etc/rc.log** file for messages that may indicate why the script hung. After troubleshooting the problem, reboot the system and see if the problem is solved.

© 2003 Hewlett-Packard Development Company, L.P.

7–10. SLIDE: Creating Custom Start Scripts

Creating Custom Startup Scripts

1. **cp /sbin/init.d/template /sbin/init.d/*myservice***
2. **vi /sbin/init.d/*myservice***
   a. Edit **start_msg** statement
   b. Edit **stop_msg** statement
   c. Edit **start** statement
      i. Change **CONTROL_VARIABLE** to ***MYSERVICE***
      ii. Add command to start your service
      iii. Add command **set_return**
   d. Edit **stop** statement
      i. Change **CONTROL_VARIABLE** to ***MYSERVICE***
      ii. Add command to stop your service
      iii. Add command **set_return**
3. **vi /etc/rc.config.d/*myservice***
   a. Add single line, ***MYSERVICE*=1**
4. **ln -s /sbin/init.d/*myservice* /sbin/rc3.d/S900*myservice***
   ln -s /sbin/init.d/*myservice* /sbin/rc2.d/K100*myservice*

Student Notes

Although most services and applications provide standard startup/shutdown scripts, it may occasionally be necessary to create a custom **/sbin/init.d** script on your system. This slide presents a cookbook approach for creating these scripts.

1. HP-UX includes a template **/sbin/init.d** startup script that you can copy, then modify for your particular service. Make a copy of the template using your service name as the new script name.

   ```
   # cp /sbin/init.d/template /sbin/init.d/myservice
   ```

2. Use your editor of choice to customize the new startup script.

   ```
   # vi /sbin/init.d/myservice
   ```


© 2003 Hewlett-Packard Development Company, L.P.

a. Scroll down to the **case** statement towards the middle of the script. Look for the following:

```
'start_msg')
   # Emit a _short_ message relating to running this script
   # with the "start" argument; this message appears as part
   # of the checklist.
   echo "Starting the <specific> subsystem"
   ;;
```

Customize the **echo** statement:

```
'start_msg')
  # Emit a _short_ message relating to running this script
  # with the "start" argument; this message appears as part
  # of the checklist.
  echo "Starting the myservice subsystem"
  ;;
```

b. Scroll down to the **stop_msg** portion of the case statement that looks like this:

```
'stop_msg')
   # Emit a _short_ message relating to running this script
   # with the "stop" argument; this message appears as part
   # of the checklist.
   echo "Stopping the <specific> subsystem"
   ;;
```

Customize this **echo** statement, too:

```
'stop_msg')
    # Emit a _short_ message relating to running this script
    # with the "stop" argument; this message appears as part
    # of the checklist.
    echo "Stopping the myservice subsystem"
    ;;
```

c. Scroll down to the start argument in the case statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   :
fi
;;
```

Customize the **CONTROL_VARIABLE** to match your service name, and add the command necessary to start the service. If you are starting a daemon that should run perpetually on your system, be sure to start it in the background. Also add a call to the **set_return** function to notify **/sbin/rc** if the daemon successfully starts:

© 2003 Hewlett-Packard Development Company, L.P.

```
# Check to see if this script is allowed to run...
if [ "$MYSERVICE" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   /opt/myservice/bin/myservice &
   set_return
   :
fi
;;
```

d. Next, scroll down to the stop argument in the case statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
fi
;;
```

Change the **CONTROL_VARIABLE**, and add the command necessary to kill the service. Some applications include a script that should be used to kill their daemons. Otherwise, just use the **`kill`** command. In either case, be sure to add a call to the **`set_return`** function to notify **`/sbin/rc`** if the daemon successfully starts:

```
# Check to see if this script is allowed to run...
if [ "$MYSERVICE" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
   kill $(ps -ef|grep /opt/myservice/bin/myservice|grep -v
grep|cut -c10-14)
   set_return
fi
;;
```

e. Save your changes and quit the editor.

Create a configuration file and a control variable for your new startup script:

```
# vi /etc/rc.config.d/myservice
  MYSERVICE=1
```

3. Create **`start`** and **`kill`** links for the new service. You may use any sequence number you wish, but the "don't care" sequence numbers (S900 and K100) are recommended.

```
# ln -s /sbin/init.d/myservice /sbin/rc3.d/S900myservice
# ln -s /sbin/init.d/myservice /sbin/rc2.d/K100myservice
```


© 2003 Hewlett-Packard Development Company, L.P.

4. Test your new startup script by executing both the start and kill links interactively. After running each script. Use **ps** to verify that the scripts succeed.

```
 # /sbin/rc3.d/S900myservice start
# ps –ef | grep myservice
# /sbin/rc2.d/K100myservice stop
# ps –ef | grep myservice
```

5. Finally, try changing run levels a few times, and watch the checklist to verify that your scripts succeed.

```
# init 2
# init 3
# init 2
```

Note that the first **init 2** may fail. Can you explain why?

© 2003 Hewlett-Packard Development Company, L.P.

7–11. LAB: Starting Network Services

Directions

Work on your own to perform the following tasks.

Part 1: Exploring the Startup/Shutdown Scripts

You have seen in this chapter that many system and network services are started automatically during the boot process via "S" scripts in the **`/sbin/rc*.d`** directories. You can view a list of these scripts by typing:

```
# ls /sbin/rc*.d/S*
```

Answer the questions below, using the output from the **`ls`** command above.

1. At which run level does NFS client functionality start?

2. At which run level does NFS server functionality start?

3. At which run level does your system set its host name?

4. At which run level does the **`net`** script set your IP address?

© 2003 Hewlett-Packard Development Company, L.P.

5. At which run level does the **`sendmail`** daemon begin delivering mail?

6. At which run level does the NIS service become available?

7. At which run level does the system enable access to **`ftp`**, **`telnet`**, and other Internet services? HINT: Internet services are started by the **`inetd`** Internet daemon.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Starting and Stopping Services

Most services may be manually started and stopped using the startup scripts in the **`/sbin/init.d`** directory.

1. Is the **`sendmail`** daemon currently running on your machine?

2. Stop the **`sendmail`** daemon using the **`init.d`** script.

3. Is the **`sendmail`** daemon running?

4. Restart **`sendmail`** properly, then check to ensure the daemon is running

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Enabling, Disabling, and Configuring Services

There are many network and system services available, but you may not need all of those services to be enabled. For instance, if you do not use networked file systems, you may choose to disable NFS. Most services may be enabled or disabled via their control variables. Usually control variables match the name of the service they control, for example, the **`sendmail`** daemon is controlled by the **`SENDMAIL`** control variable.

Setting a control variable to "1" enables that service at next boot, while setting the control variable to "0" disables the service at next boot. Control variables are set in configuration files in **`/etc/rc.config.d/*`**. Sometimes the configuration file matches the name of the service. You can always use the **`grep`** command to find the proper configuration file for a service. For instance, the output from the following **`grep`** command suggests that the sendmail control variable is defined in **`/etc/rc.config.d/mailservs`**.

```
# grep -il sendmail /etc/rc.config.d/*
/etc/rc.config.d/mailservs
```

See if you can find the **`/etc/rc.config.d`** configuration files for each of the services below, and determine which of those services are enabled on your system.

Service Name	Configuration File Name	Enabled?
`nfs.server`		
`nfs.client`		
`nis.server`		
`nis.client`		
`sendmail`		
`named` (DNS)		
`xntpd`		

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Creating a Custom Startup Script

In this part of the lab exercise, you will have an opportunity to create a custom startup/shutdown script to start and stop the **pfs_mountd** daemon used by the PFS file system in HP-UX. The "Portable File System" is one of the few services in HP-UX that does not include a pre-configured startup script, so this is a particularly practical exercise!

1. Make a copy of the **/sbin/init.d/template** to use as a template for your **pfs_mountd** startup script.

   ```
   # cp /sbin/init.d/template /sbin/init.d/pfs_mountd
   ```

2. Use your editor of choice to customize the new startup script.

   ```
   # vi /sbin/init.d/pfs_mountd
   ```

   a. Scroll down to the **case** statement towards the middle of the script. Look for the following:

   ```
   'start_msg')
         # Emit a _short_ message relating to running this script
         # with the "start" argument; this message appears as part
         # of the checklist.
          echo "Starting the <specific> subsystem"
          ;;
   ```

   Change the **echo** statement to the following:

   ```
   'start_msg')
       # Emit a _short_ message relating to running this script
       # with the "start" argument; this message appears as part
       #  of the checklist.
       echo "Starting the pfs_mountd subsystem"
       ;;
   ```

   b. Scroll down to the **stop_msg** portion of the **case** statement that looks like this:

   ```
   'stop_msg')
       # Emit a _short_ message relating to running this script
       # with the "stop" argument; this message appears as part
       # of the checklist.
       echo "Stopping the <specific> subsystem"
       ;;
   ```

   Change the **echo** statement to the following:

   ```
   'stop_msg')
       # Emit a _short_ message relating to running this script
       # with the "stop" argument; this message appears as part
       # of the checklist.
       echo "Stopping the pfs_mountd subsystem"
       ;;
   ```


© 2003 Hewlett-Packard Development Company, L.P.

c. Scroll down to the **start** argument in the **case** statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   :
fi
;;
```

Change the **CONTROL_VARIABLE**, and add the command necessary to start **pfs_mountd** as shown below. Also add a call to the **set_return** function to notify **/sbin/rc** if the daemon successfully starts:

```
# Check to see if this script is allowed to run...
if [ "$PFS_MOUNTD" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   /usr/sbin/pfs_mountd &
   set_return
   :
fi
;;
```

d. Next, scroll down to the `stop` argument in the `case` statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
fi
;;
```

f. Change the **CONTROL_VARIABLE**, and add the command necessary to kill **pfs_mountd** as shown below. Also add a call to the **set_return** function to notify **/sbin/rc** if the daemon successfully starts:

```
# Check to see if this script is allowed to run...
if [ "$PFS_MOUNTD" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
   kill $(ps -ef | grep /usr/sbin/pfs_mountd | grep -v grep |\
   cut -c10-14)
   set_return
fi
;;
```

e. Save your changes and quit **/sbin/init.d/pfs_mountd**.

© 2003 Hewlett-Packard Development Company, L.P.

3. Create a configuration file and a control variable for your new startup script:

```
# vi /etc/rc.config.d/pfs_mountd
  PFS_MOUNTD=1
```

4. Create a start link to start the new service at run level 3 using the "don't care" 900 sequence number, and a kill link to kill the new service with sequence number 100 at run level 2:

```
# ln -s /sbin/init.d/pfs_mountd /sbin/rc3.d/S900pfs_mountd
# ln -s /sbin/init.d/pfs_mountd /sbin/rc2.d/K100pfs_mountd
```

5. Test your new startup script by executing both the start and kill links.

```
# /sbin/rc3.d/S900pfs_mountd start
# ps -ef | grep pfs_mountd
# /sbin/rc2.d/K100pfs_mountd stop
# ps -e
```

6. Assuming the previous test succeeded, try changing run levels a few times to further test your scripts.

```
# init 2
# init 3
# init 2
```

   Note that the first **`init 2`** may fail. Can you explain why?

© 2003 Hewlett-Packard Development Company, L.P.

Module 8 — NFS Concepts

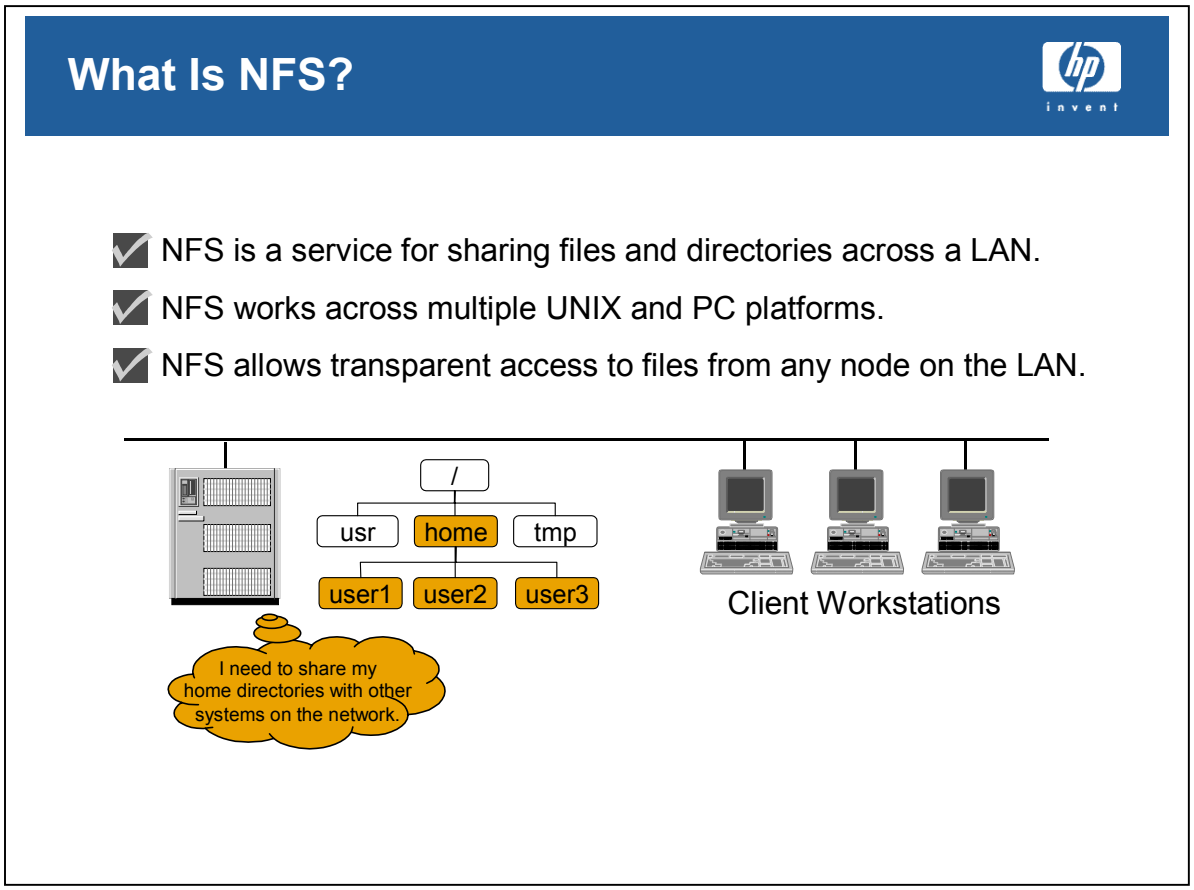
Objectives

Upon completion of this module, you will be able to do the following:

- Describe the purpose and function of NFS.
- Define **NFS server** and **NFS client**.
- List probable candidates for file sharing via NFS.
- Describe the purpose of NFS RPCs.
- Describe the purpose of the **`portmap`** and **`rpcbind`** daemons.
- Compare and contrast the NFS PV2 and NFS PV3 protocols.
- Compare and contrast the NFS and CIFS protocols.

© 2003 Hewlett-Packard Development Company, L.P.

8–1. SLIDE: What Is NFS?



Student Notes

- NFS is a service for sharing files and directories across a LAN.

  The first module in this course noted that the primary purpose of a LAN is to provide a mechanism for sharing resources. Disk space is one of the most commonly shared resources on LANs today. Although many file sharing solutions have been developed over the years, Sun's Network File System (NFS) protocol is by far the most common in the UNIX world today. Using NFS, administrators can share executables, data files, and even home directories across multiple systems on Local- and Wide-Area Networks.

- NFS works across multiple UNIX and PC platforms.

  NFS was first released by Sun in the early 1980s and was ported to HP-UX in 1986. Today, nearly every UNIX platform available supports NFS. In fact, the client portion of NFS has even been ported to the Microsoft and Macintosh operating systems! File systems shared from an HP-UX NFS server can be mounted on any one of these NFS clients.

© 2003 Hewlett-Packard Development Company, L.P.

- NFS allows transparent access to files from any node on the LAN.

  NFS is virtually transparent to users and applications on the NFS clients. The same file manipulation commands (**`cp`**, **`mv`**, **`ls`**, **`cat`**, and so on) and system calls (**`open()`**, **`write()`**, **`read()`**, and so on) that are used to access files on a local HFS or VxFS file system can also be used to access files on an NFS file system. When users **`cd`** to **`/home/user1`**, they may be accessing a directory physically stored on a local logical volume, or on a disk attached to an NFS server elsewhere on the network.

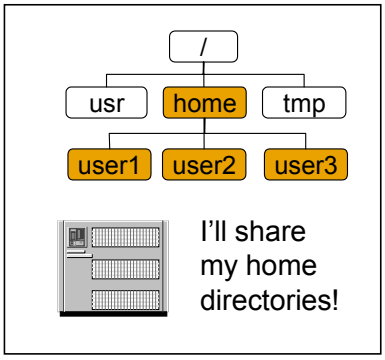
The remainder of this chapter introduces some key NFS concepts and terminology, while the next two chapters discuss NFS configuration issues.

© 2003 Hewlett-Packard Development Company, L.P.

8–2. SLIDE: What Files Should I Share via NFS?

What Files Should I Share via NFS?

Good candidates for file sharing via NFS:
- Home directories
- Application files under `/opt`
- Operating System files under `/usr`
- Data files used by multiple nodes



Poor candidates for file sharing via NFS:
- Device files under `/dev`
- System-specific configuration files under `/etc`
- Dynamic operating system files under `/var`
- Single-user mode command files under `/sbin`

Student Notes

NFS can be used to share almost any file on an HP-UX system. However, some files and directories are better candidates than others.

Good Candidates for File Sharing via NFS

- Storing home directories on an NFS server offers many advantages. Users can log in on any workstation on the LAN and have access to their home directory. Administrators are saved the drudgery of scheduling backups on individual workstations if users store all their files on a central server. Disk space management is simplified since users store files on the server rather than their local disks. However, there are disadvantages to this approach. If the server goes down, users will be able to login, but will be placed in the **/** directory rather than their normal home directories. Storing home directories on an NFS server may also dramatically increase network traffic. The **root** home directory should always be stored in a local file system to ensure that it is available even when the network is inaccessible.

- Application directories under **/opt** can be stored on the NFS server. Doing so provides a central point of administration and saves disk space on users' desktop machines. If you choose to share executables via NFS, make sure you do not mount a file system full of

© 2003 Hewlett-Packard Development Company, L.P.

Solaris executables on your HP-UX box, or vice-versa! Although NFS provides transparent access to files across platforms, the code contained in those files may be platform-specific!

- When disk space was more expensive, some administrators stored the **`/usr/lib`**, **`/usr/share`**, **`/usr/local`**, and **`/usr/contrib`** on NFS servers. As disks have become cheaper, most administrators have chosen to store these directories on users' local disks to minimize network traffic.
- Data files shared by multiple nodes are ideal candidates for sharing via NFS, too.

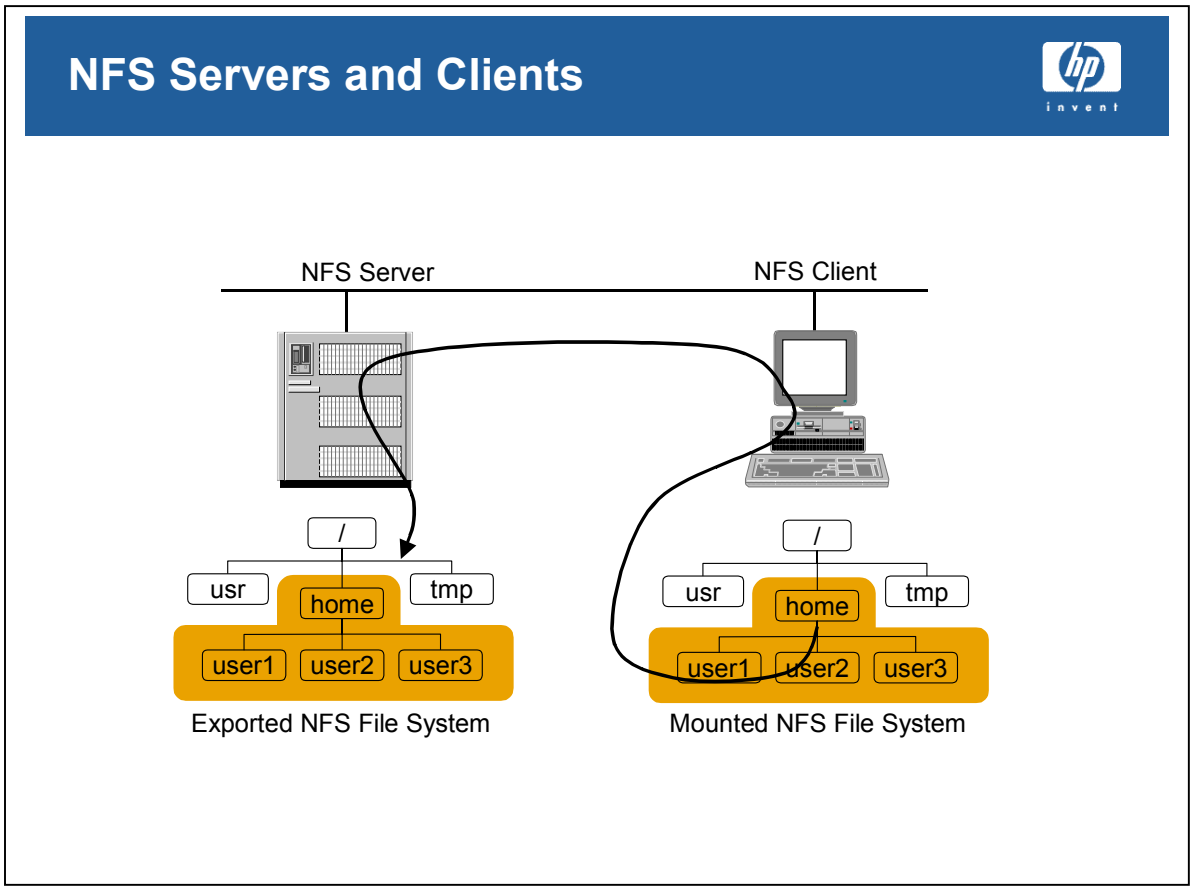
Poor Candidates for File Sharing via NFS

Generally speaking, host-specific files should not be shared or mounted via NFS.

- Device files under **`/dev`** are certainly host specific.
- System-specific configuration files under **`/etc`** should not be shared via NFS.
- With the exception of the email directory, **`/var/mail`**, **`/var`** is rarely shared.
- **`/sbin`** contains executables used in the early stages of the boot process. Since these programs run before network connectivity is established, **`/sbin`** should always be stored on a local disk.

© 2003 Hewlett-Packard Development Company, L.P.

8–3. SLIDE: NFS Servers and Clients



Student Notes

Hosts in an NFS environment can be configured as NFS servers, NFS clients, or both.

NFS Servers

A host on which a shared file system physically resides is known as an NFS server. The NFS server administrator can choose which directories and files should be made available to other hosts.

- The administrator can choose to share an entire file system, such as **`/home`**, or **`/opt`**.
- The administrator can choose to share only one or more subdirectories within a file system. For instance, instead of sharing the entire **`/home`** file system, the administrator can simply choose to share the **`/home/user1`** and **`/home/user2`** subdirectories.
- The administrator can even choose to share a single file, such as **`/home/user1/data`**!

File systems, directories, and files that have been made available to other hosts via NFS are said to be "exported."

© 2003 Hewlett-Packard Development Company, L.P.

NFS Clients

Hosts that access NFS file systems from an NFS server are called NFS clients. NFS file systems must be mounted on a local mount point directory in much the same way that a local logical volume is mounted on a mount point directory. After an NFS file system is mounted on a mount point directory, all attempts to access files and directories below that mount point are automatically forwarded to the NFS server.

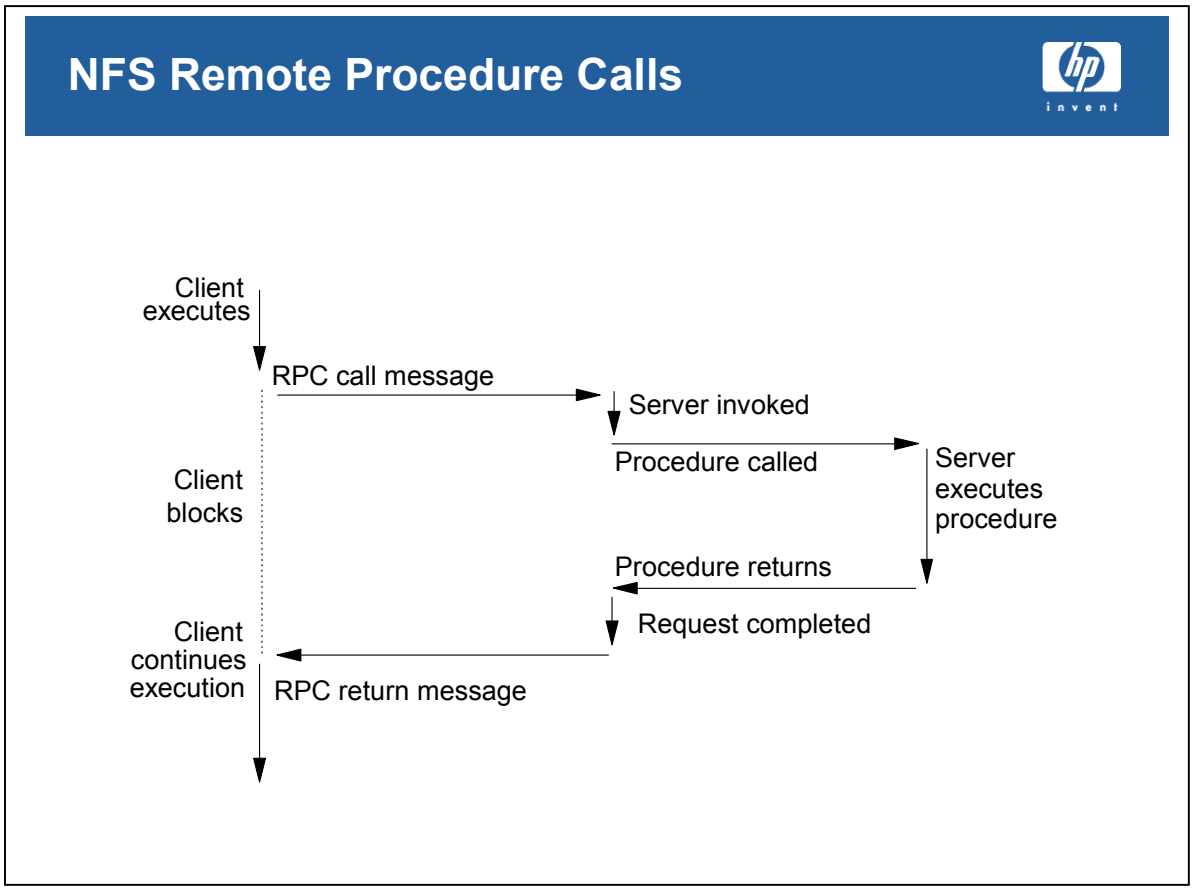
The NFS client administrator may choose to mount all or part of an exported file system. For instance, if the NFS server administrator exports **`/home`**, the client administrator may choose to mount the entire **`/home`** file system via NFS, or a single subdirectory from within **`/home`**.

Dual Role Hosts

A default HP-UX install actually enables both NFS server *and* client functionality. It is perfectly acceptable for a host to mount a file system from an NFS server, and then export a different file system to other NFS clients. However, it is not possible for a host to mount an NFS file system from a server, and then re-export that same file system to other NFS clients.

© 2003 Hewlett-Packard Development Company, L.P.

8–4. SLIDE: NFS Remote Procedure Calls



Student Notes

The NFS remote mount capability is implemented via "Remote Procedure Calls" (RPCs) developed by Sun Microsystems.

The RPC mechanism makes it possible for a client system to execute a procedure remotely on an NFS server. Most of the system calls that applications use to access local file systems have closely related RPC calls. For instance, applications use the **`read()`** system call to read from a file; NFS clients use a **`read()`** RPC to read from a file on an NFS server. Applications use the **`write()`** system call to write data to a local file; NFS clients use a **`write()`** RPC to write data to a file stored on an NFS server. These are just a couple of the RPCs recognized by an NFS server.

When an application executes a file access system call, the kernel automatically determines if the target file is on a local device that can be accessed directly, or an NFS file system that may require an RPC call. If the target file is on an NFS file system, the client's kernel automatically sends an appropriate RPC request to the NFS server. Thus, NFS is transparent to your applications and processes.

© 2003 Hewlett-Packard Development Company, L.P.

Other important points regarding RPCs:

- RPCs are designed to be platform independent. Windows, Macintosh, and UNIX clients all use the same RPC requests to access NFS servers.

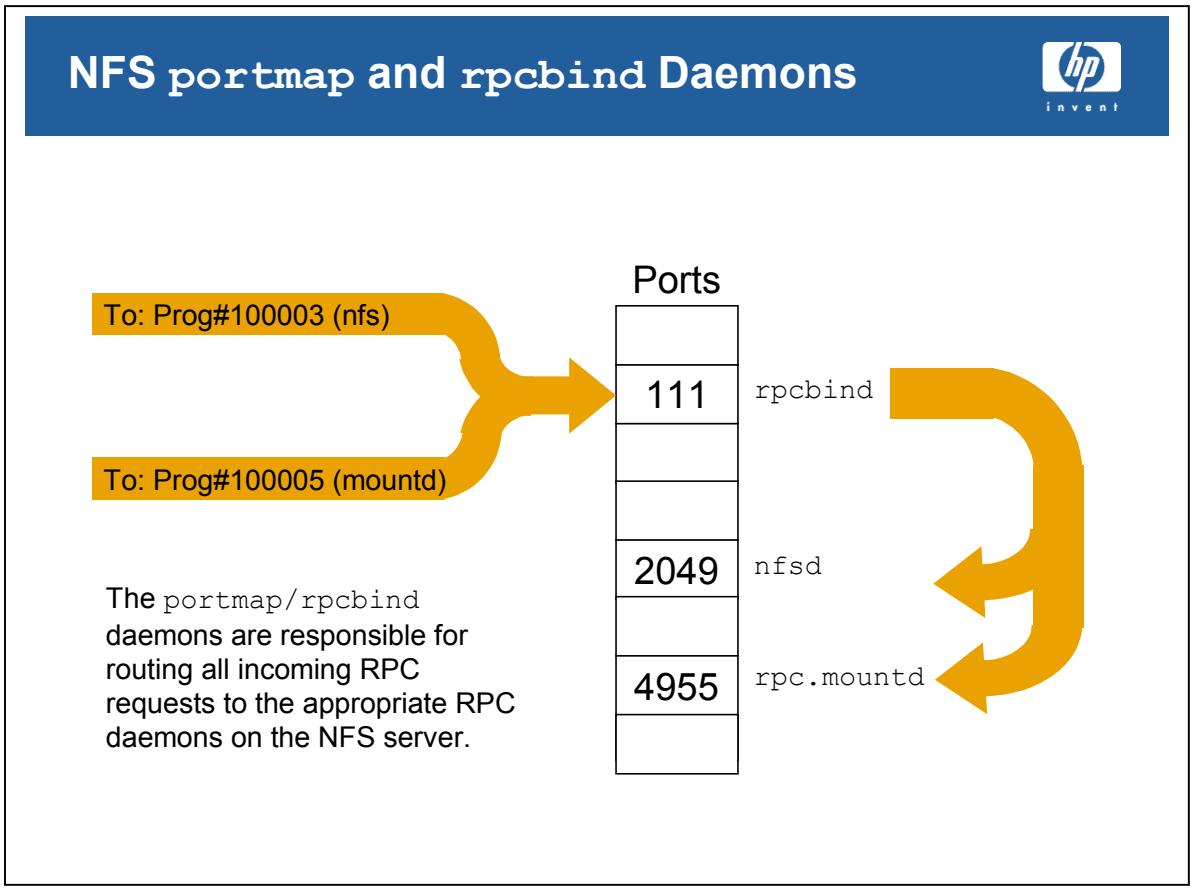
- Each RPC takes one parameter and returns one result.

- All data passed to and from RPC procedures is encoded using a platform-independent format called the External Data Representation (XDR) standard. This makes it possible for hosts using different byte ordering, size, and word alignments to pass data back and forth successfully.

- Although NFS is the most common service that uses Sun's remote procedure calls, other services, such as NIS, use RPCs, too.

© 2003 Hewlett-Packard Development Company, L.P.

8–5. SLIDE: NFS `portmap` and `rpcbind` Daemons



Student Notes

RPCs use sockets and the TCP/UDP transport protocols to pass data between NFS clients and servers. At boot time, the NFS server launches several RPC programs to handle incoming RPC requests from clients. Each RPC program listens for requests on a separate, randomly chosen port number.

If the RPC programs listen for incoming requests on randomly chosen port numbers, how do the clients know to which port number to address their requests? When the RPC programs start up, the **`rpcbind`** daemon registers which RPC programs are running on which ports. RPC clients simply send their RPC requests to the **`rpcbind`** daemon, which always runs on port number 111. **`rpcbind`** then forwards the incoming RPC requests to the appropriate port numbers.

Clients specify the RPC program they wish to contact by "Program Number." The **`/etc/rpc`** file associates RPC programs with their well-known program numbers. Although an RPC program's port number may vary from system to system, and reboot to reboot, the RPC program numbers are consistent across all platforms and hosts. This ensures that Solaris NFS clients can successfully communicate with HP-UX NFS servers, and vice versa.

© 2003 Hewlett-Packard Development Company, L.P.

This mechanism for dynamically binding RPC programs to port numbers is desirable because the range of reserved port numbers is very small, and the number of potential RPC programs is very large.

Starting and Stopping `rpcbind`

If the **`rpcbind`** daemon crashes, all RPC server daemons must be restarted so they can be re-registered.

If **`rpcbind`** aborts or terminates on SIGINT or SIGTERM, it will write the current list of registered services to **`/tmp/portmap`** and **`/tmp/rpcbind.file`**. Starting **`rpcbind`** with the **`-w`** option instructs it to look for these files and start operation with the registrations found in them. This allows **`rpcbind`** to resume operation without requiring all RPC services to be restarted.

CAUTION: The **`rpcbind`** daemon must be started before **`inetd`**.

WARNING: **If `rpcbind` crashes, all RPC server daemons must be restarted.**

A Note for 10.20 Administrators

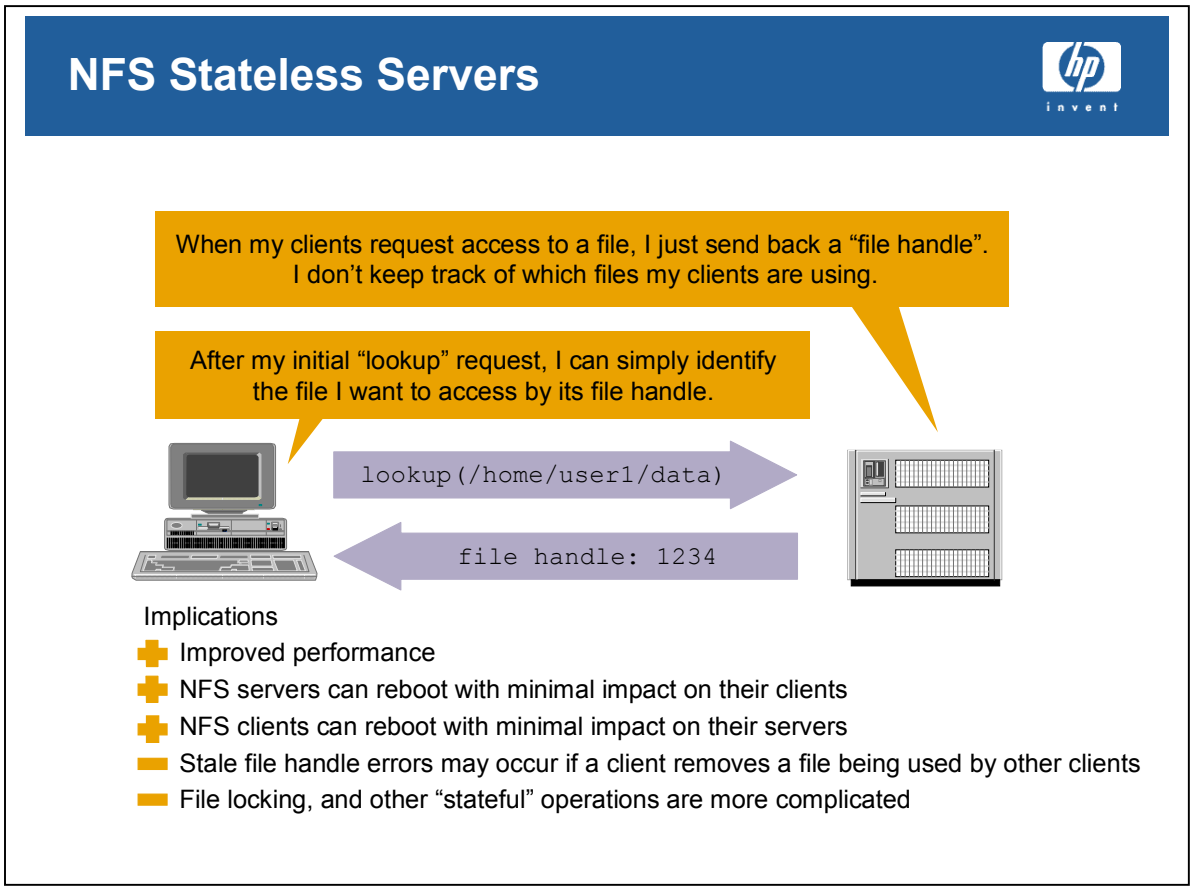
Before HP-UX 11.00, the **`portmap`** daemon served the purpose that is currently served by **`rpcbind`**. The two daemons are indistinguishable to your users and applications.

Example `/etc/rpc`

```
##
#  file of rpc program name to number mappings
##

portmapper     100000 portmap sunrpc
nfs            100003 nfsprog
mountd         100005 mount showmount
pcnfsd         150001 pcnfs
llockmgr       100020
nlockmgr       100021
status         100024
  :
```


© 2003 Hewlett-Packard Development Company, L.P.

8–6. SLIDE: NFS Stateless Servers



Student Notes

One key difference between NFS and local disk-based file systems is that NFS operates in a "stateless" manner, while local file systems operate in a "statefull" manner.

When applications open files on a local disk-based file system, the kernel uses "file descriptors" to track which processes are using which files. When a user removes a file from a local file system, the file's data blocks are not actually de-allocated until the last user using the file is finished. Similarly, if the administrator attempts to unmount a local file system that is still being used by a user, the **`umount`** command fails with a "device busy" message. In other words, local file systems are accessed in a "statefull" manner; the kernel tracks which files and directories are being used by whom, and prevents one user's requests from interfering with others' requests.

NFS, on the other hand, operates in a "stateless" manner. When a client opens a file on an NFS server via the **`lookup()`** RPC, the server sends the client a "file handle" derived from the requested file's inode number. The server does not record the fact that the file is in use, nor does it create a file descriptor to record which portion of the file the client is currently accessing. Since the server does not maintain state, a client may possibly remove a file that another client still has open for reading. An NFS client can even remove another client's

© 2003 Hewlett-Packard Development Company, L.P.

present working directory! Both of these situations result in "stale file handles": file handles that reference files or directories that no longer exist.

NFS statelessness has both advantages and disadvantages:

- Advantage: Improved performance. Maintaining client state information would place a heavy burden on NFS servers.

- Advantage: NFS servers can reboot with minimal impact on their clients. After a reboot, NFS servers can immediately resume processing as if nothing had happened. Client file handles should remain unchanged, and each client simply re-transmits any access requests that went unanswered while the server was down. If NFS were a statefull protocol, some sort of complicated recovery process would be required to determine which clients had files open at the time of the reboot.

- Advantage: NFS clients can reboot with minimal impact on their servers. Since the server does not attempt to track which clients have open files, a downed client requires no action on the part of the server.

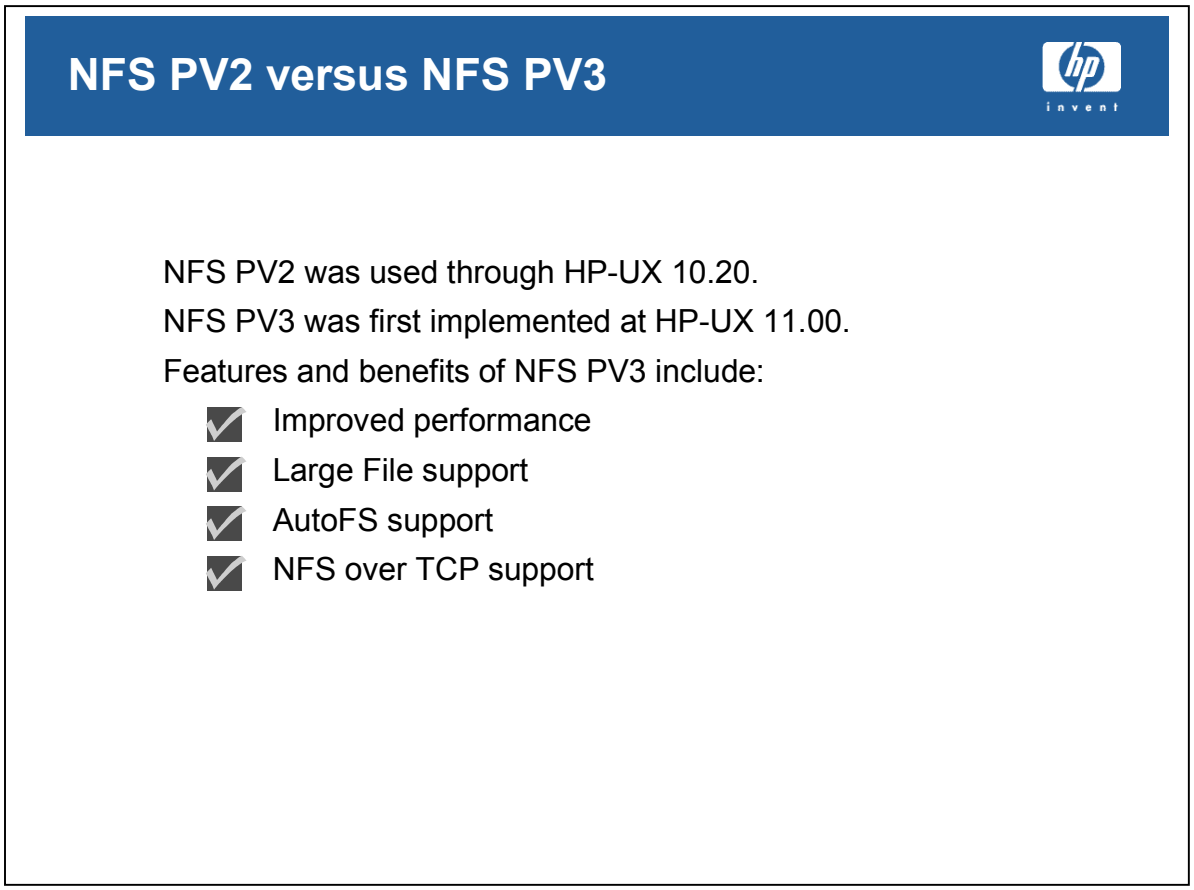
- Disadvantage: Stale file-handle errors may occur if a client removes a file being used by other clients. Since the NFS server does not attempt to track which files are being used by its NFS clients, NFS allows clients to remove files that are still in use by other clients.

- Disadvantage: File locking and other “stateful” operations are more complicated. Some applications use file locks to ensure that only one process at a time may access critical files. Since NFS does not track which files are in-use, file locking becomes more complicated. File locking is, however, possible via two daemons that are included with NFS: **`rpc.lockd`** and **`rpc.statd`**. Clients that wish to lock a region of a file may send a request to the server's **`rpc.lockd`** daemon. **`rpc.lockd`** uses a "semaphore" to mark the requested file region "locked.” The server's **`rpc.statd`** daemon begins polling the client at regular intervals; if the client reboots unexpectedly, the server removes the lock so other clients can access the file.

  NFS only implements "advisory" locks. When an application attempts to access a file, the onus is on the application to check for existing advisory locks on the file; NFS does not forcefully prevent other processes from accessing a locked file region.

© 2003 Hewlett-Packard Development Company, L.P.

8–7. SLIDE: NFS PV2 versus NFS PV3



Student Notes

HP supports two different NFS protocol versions. HP-UX version 10.20 supported NFS Protocol Version 2 (PV2). HP-UX version 11.00 introduced support for NFS Protocol Version 3 (PV3), but retained backward compatibility with PV2. Servers running PV3 still accept mount requests from PV2 clients, and PV3 clients can still successfully mount file systems from PV2 servers. Some PV3 features have been back-ported to HP-UX 10.20.

NFS PV3 Features

- Improved performance. The NFS caching algorithms were enhanced for PV3, which may lead to significant performance gains in some environments.

- Large file support. One of the most beneficial features of NFS PV3 is its ability to support large files. NFS Version 2 supported a 32-bit file size, while NFS Version 3 supports a 64-bit file size. The maximum file size on NFS PV2 is only 2 Gigabytes, while NFS PV3 supports a maximum file size of 128 Gigabytes.

- AutoFS support. NFS PV2 included a service called "automounter," which automatically mounted and unmounted NFS file systems on an as-needed basis. NFS PV3 includes a

© 2003 Hewlett-Packard Development Company, L.P.

more flexible, more robust version of automounter called AutoFS. Automounter and AutoFS will be discussed in detail later in the course.

- NFS over TCP support. NFS PV2 and the initial release of NFS PV3 used the UDP protocol to transmit RPC traffic between NFS servers and clients. UDP functions well on local area networks, but often generates excessive timeouts and other performance problems on wide area networks. In February 2000, HP released a patch for 11.0 NFS PV3 that supports NFS over TCP (see the text below for details). TCP is the default NFS transport protocol at HP-UX 11i. The NFS over TCP functionality is not available for HP-UX 10.20.

Enabling NFS over TCP on HP-UX 11.00

TCP is the default NFS transport protocol at HP-UX 11i, but must be manually enabled on HP-UX 11.00 via the following procedure:

1. Look on the **`http://www.itrc.hp.com`** website for the latest 11.00 NFS over TCP patch. Install the patch and all its dependencies according the **`.text`** file included with the patch.

2. Reboot your system.

3. Add the **`NFS_TCP`** variable to the bottom of the **`/etc/rc.config.d/nfsconf`** file:

```
# vi /etc/rc.config.d/nfsconf
  NFS_TCP=1
```

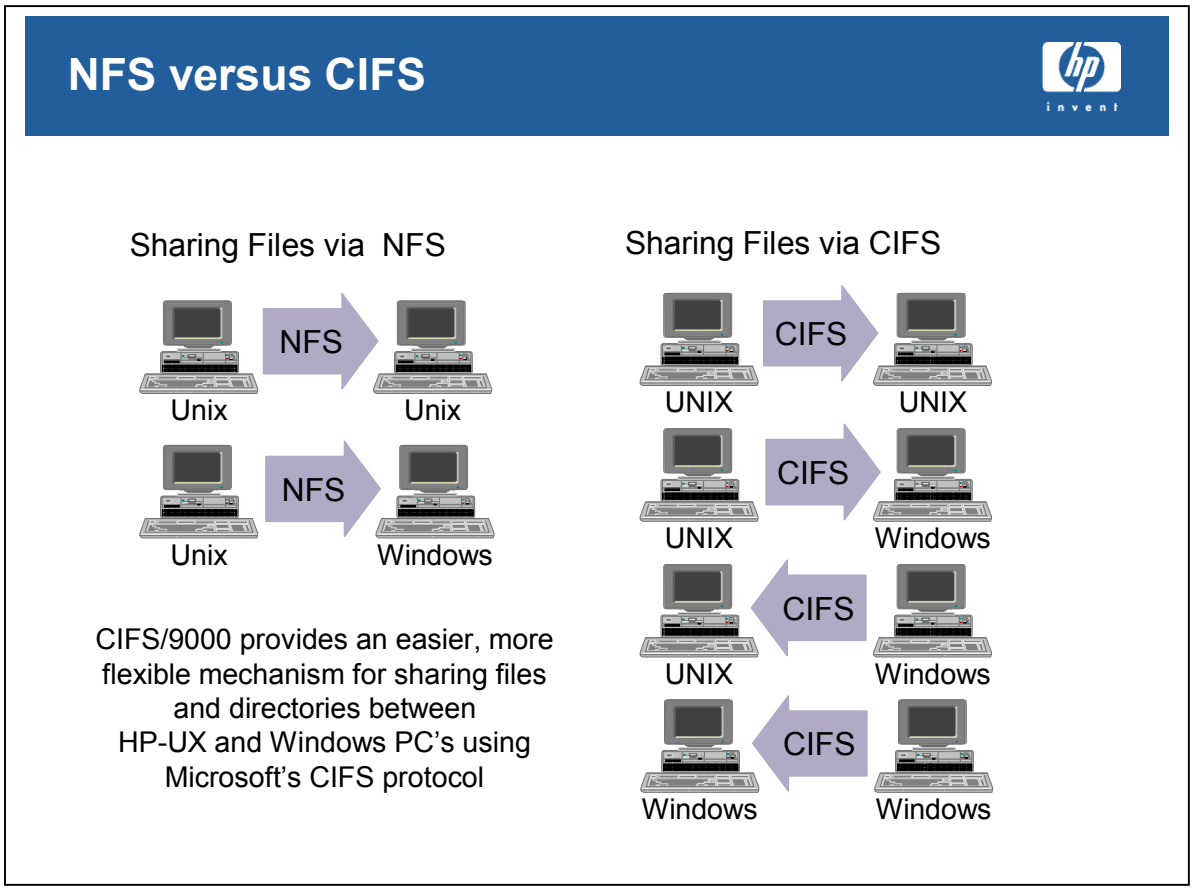
4. Stop and restart NFS.

```
# /sbin/init.d/nfs.server stop
# /sbin/init.d/nfs.client stop
# /sbin/init.d/nfs.client start
# /sbin/init.d/nfs.server start
```

After going through this procedure, your host will attempt to use TCP whenever possible. If a server or client does not support NFS over TCP, your host will automatically revert to NFS over UDP.

© 2003 Hewlett-Packard Development Company, L.P.

8–8. SLIDE: NFS versus CIFS



Student Notes

NFS is the de facto standard for file sharing among UNIX systems, and NFS client functionality has even been ported to the Microsoft Windows. However, since NFS is not a native Windows protocol, an NFS server does not provide all of the functionality provided by a regular Windows NT file server:

- NFS servers cannot provide Windows Primary Domain Controller functionality.
- NFS servers cannot provide Windows Name Resolution Services (WINS).
- NFS file systems do not appear in Windows clients' network neighborhood browsers.

Finally, NFS provides no functionality for exporting Windows file systems back to UNIX clients.

CIFS/9000

Now there is an alternative for administrators who wish to share file and print services in a heterogeneous environment. HP-UX 11.x supports a new product called CIFS/9000 that provides a full implementation of Microsoft's "Common Internet File System" protocol, which is used by Windows 95, Windows 98, Windows 2000, and NT for sharing file and printer resources. Using CIFS/9000, HP-UX, and Microsoft Windows systems can seamlessly and transparently share resources.

© 2003 Hewlett-Packard Development Company, L.P.

CIFS/9000 includes several components:

- The server portion of CIFS/9000 is based on Samba, an open source CIFS server solution that has been ported to many UNIX platforms. File systems made available from an HP-UX box via Samba can be mounted on Windows clients as standard drive letters and can be accessed via the Windows "Network Neighborhood" and "Windows Explorer" like standard Microsoft file shares. In fact, your HP-UX Samba server can even be a Primary Domain Controller and print server for Microsoft clients!

- HP includes CIFS client software in the CIFS/9000 product. This software makes it possible to mount file shares from any Samba or Microsoft server on an HP-UX client using the **`/etc/fstab`** file and the standard UNIX **`mount`** command. File systems mounted via the CIFS client software may be accessed using all the standard UNIX utilities and system calls.

- Finally, the CIFS/9000 product includes a Pluggable Authentication Module (PAM) library to allow users to log onto their HP-UX systems using their Windows domain usernames and passwords.

CIFS/9000 is not available for HP-UX 10.x but is included for no additional charge on the HP-UX 11.x Applications CD.

The remaining notes on this slide describe the steps required to configure a simple CIFS server and client. For more information on Samba and CIFS, sign up for one of HP's UNIX/NT integration courses, read HP's CIFS documentation on **`http://docs.hp.com`**, or purchase O'Reilly and Associates, *Using Samba* (ISBN 1-56592-449-5).

Configuring a Simple CIFS/9000 Server

1. Install the CIFS/9000 server bundle from the HP-UX 11.x Applications CD.

   ```
   # mkdir /cdrom
   # mount /dev/dsk/cxtxdx /cdrom  #use your CDROM's device file
   # swinstall -s /cdrom
   ```

2. Configure the **SAMBA** control variable to enable the Samba daemons after every reboot.

   ```
   # vi /etc/rc.config.d/samba
     RUN_SAMBA=1
   ```

3. Create or modify the **/etc/opt/samba/smb.conf** configuration file to specify which files and directories you want to share with CIFS clients. You may edit this file with **vi**, or use the **/opt/samba/bin/swat** GUI based configuration tool. The sample file below exports all user home directories and the **/tmp** directory. Note that there are over a hundred parameters that may be specified in the **smb.conf** file. This sample file lists only the most basic parameters required to share a few directories.

   Replace the ***hostname*** parameter with your server's hostname. Replace the ***WORKGROUP*** parameter with your clients' workgroup name or NT domain name. Replace the ***128.1.*** parameter with a space separated list of subnets that need access to the shares on this server.

© 2003 Hewlett-Packard Development Company, L.P.

```
# vi /etc/opt/samba/smb.conf
  [global]
    netbios name      = hostname
    workgroup         = WORKGROUP
    server string     = Samba Server
    hosts allow       = 128.1.
    security          = user
    encrypt passwords = yes
  [homes]
    comment           = Home Directories
    writeable         = yes
    browseable        = yes
  [tmp]
    comment           = Temporary Directory
    path              = /tmp
    writeable         = yes
    browseable        = yes
```

4. Run the Samba **testparm** program to search for syntax errors in your configuration file. This will also list all of the default parameters that will be set for you automatically.

```
# /opt/samba/bin/testparm
```

5. Create a Samba password file. This file determines which client users will be able to access your CIFS shared directories.

```
# touch /var/opt/samba/private/smbpasswd
# chmod 500 /var/opt/samba/private
# chmod 600 /var/opt/samba/private/smbpasswd
```

6. Add a few of the users from your UNIX password file to the Samba password file. The usernames specified must already exist in the **/etc/passwd** file.

```
# /opt/samba/bin/smbpasswd -a user1
```

7. Start the Samba daemon.

```
# /sbin/init.d/samba start
```

8. Use the **smbclient** utility to verify that your Windows domain/workgroup and username are set properly and to list the shares that have been made available to clients. You can replace the "**%**" sign with a specific username if you wish to see which shares are available for a specific Windows user.

```
# /opt/samba/bin/smbclient -L localhost -U%
```


© 2003 Hewlett-Packard Development Company, L.P.

Configuring a CIFS/9000 Client

1. Install the CIFS/9000 Client bundle from the HP-UX 11.x Applications CD.

```
# mkdir /cdrom
# mount /dev/dsk/cxtxdx /cdrom  #use your CDROM's device file
# swinstall -s /cdrom
```

2. Define your Windows workgroup or domain name in the **cifsclient.cfg** file.

```
# vi /etc/opt/cifsclient/cifsclient.cfg
  domain = "WORKGROUP"
```

3. Configure the **RUN_CIFSCLIENT** variable to ensure that the client daemon starts after every system boot, then run the startup daemon to start the daemon.

```
# vi/etc/rc.config.d/cifsclient
RUN_CIFSCLIENT=1
# /sbin/init.d/cifsclient start
```

4. Create mount point directories for your CIFS file system(s).

```
# mkdir /homes
```

5. Add the CIFS file system(s) to your **/etc/fstab** file. (Replace "**server**" with your Samba server's hostname.)

```
# vi /etc/fstab
  server:/homes /homes cifs defaults 0 0
```

6. Mount the new CIFS file systems. If you choose to use CIFS on a production box, you would probably include this **mount** command in the same startup script that you use to execute the **cifsclient start** command.

```
# mount -aF cifs
```

7. CIFS behaves somewhat differently than NFS. Once an NFS file system is mounted, any user on the system can access that file system. In CIFS, access to file shares is granted on a user-by-user basis. Thus, even though you have already mounted your CIFS file systems, users cannot access those mounted file systems without providing a valid CIFS password. Log in as a CIFS user using one of the usernames and passwords you added to the **smbpasswd** file on the server.

```
# /opt/cifsclient/bin/cifslogin server user1
```

8. List the CIFS shares to which you have access now that you are logged in. Explore one of the shares with the **cd** and **ls** commands.

```
# cifslist -A
# ls /homes
```


© 2003 Hewlett-Packard Development Company, L.P.

9. When you are done with the CIFS file systems, terminate your connection to the CIFS server with the **cifslogout** command. Then unmount the CIFS file systems.

```
# /opt/cifsclient/bin/cifslogout server
# umount -aF cifs
```

Accessing a CIFS/9000 File System from a Windows NT Client

1. Login as any user on an NT workstation.

2. Verify that you are a member of the same workgroup as your SAMBA server.

```
Start -> Settings -> Control Panel -> Network -> Identification
```

3. Launch the Network Neighborhood tool from the Desktop, an icon should appear for your SAMBA server's hostname. Double click on the SAMBA server icon.

4. A username dialog box should pop up. Enter one of the usernames and passwords that you created on the SAMBA server. When you click **OK**, your SAMBA server shares should appear!

© 2003 Hewlett-Packard Development Company, L.P.

Module 9 — Configuring NFS

Objectives

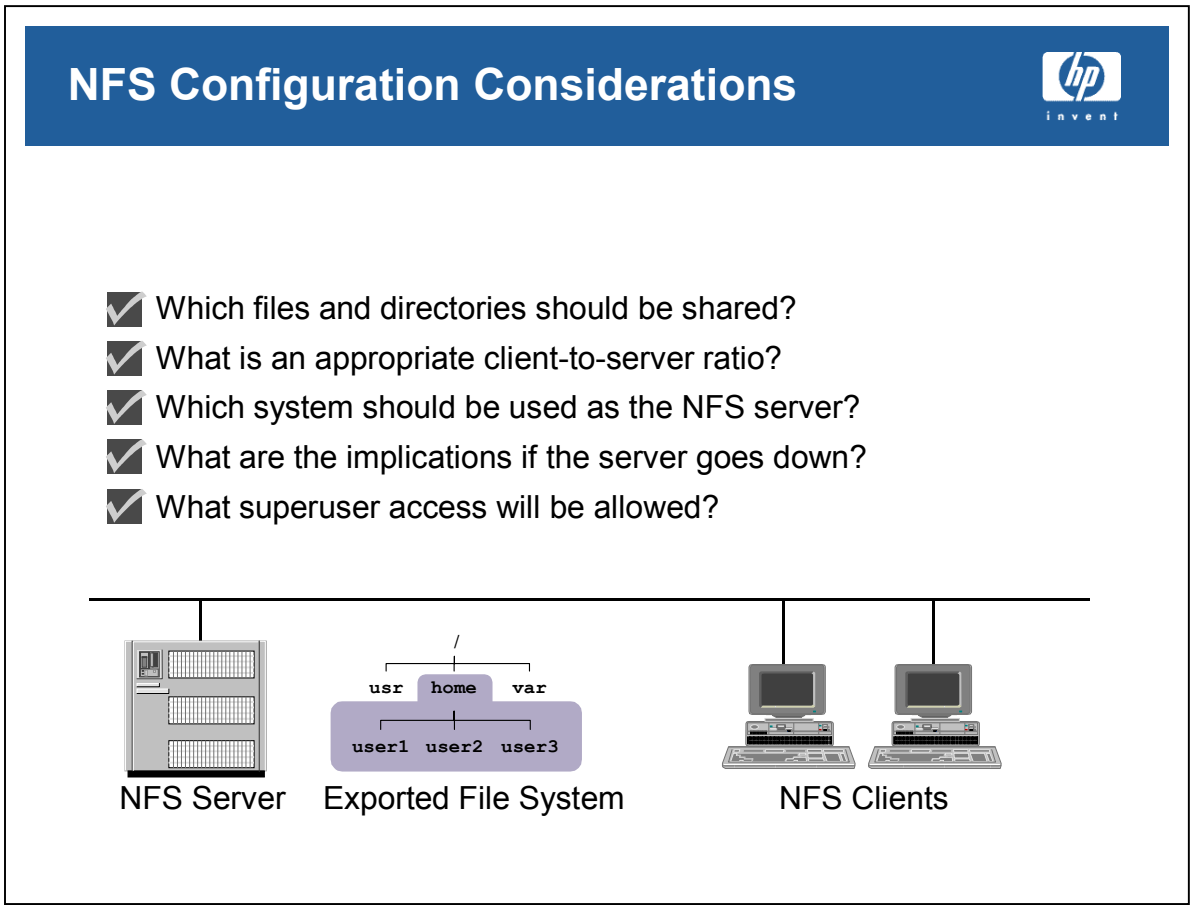
Upon completion of this module, you will be able to do the following:

- Configure NFS server functionality.
- Export file systems and determine access privileges for those file systems.
- Configure NFS client functionality.
- Mount and unmount NFS file systems.
- Automatically mount NFS file systems.
- Determine which file systems have been exported and mounted.
- Describe the function of the following NFS configuration files:

  `/etc/rc.config.d/nfsconf`
  `/etc/exports`
- List the daemons that must be running on an NFS server and client.
- Use **`showmount`**, **`rpcinfo`**, and **`nfsstat`** to troubleshoot problems with NFS.

© 2003 Hewlett-Packard Development Company, L.P.

9–1. SLIDE: NFS Configuration Considerations



Student Notes

If you decide to implement NFS, the first step is to decide exactly which file systems should be shared. The slide above highlights several issues you should consider.

- Which files and directories should be shared? Do you want to manage home directories, executable directories, data directories, or all of the above? Remember that disk-based file systems generally provide better performance than NFS file systems. Also, note that NFS can place a tremendous strain on your network infrastructure. The more file systems you share via NFS, the greater the load NFS will place on your NFS servers and network infrastructure.

- What is the client-to-server ratio? Generally speaking, as the number of NFS clients increases, the load on the NFS server grows. If you have many clients, it may be necessary to configure multiple NFS servers to share the load. The characteristics of your applications should be considered when making this decision. If the application tends to be disk-use intensive, and performance is important, you should aim for a lower client-to-server ratio. If the application is less disk-intensive, it may be possible for many more clients to share the same server.

ã 2003 Hewlett-Packard Development Company, L.P.

- Which system should be used as the NFS server? Ideally, choose the biggest, fastest system you have to be your NFS server. An underpowered NFS server may prove to be a bottleneck for all of the NFS clients. Your HP Sales representative should be able to help you size your NFS server appropriately.

- What are the implications if the server goes down? NFS provides a single point of administration; however, that single point of administration becomes a single point of failure if the NFS server crashes! If the NFS server does go down, what impact will that have on your clients? If all of your users' home directories are stored on the NFS server, no clients will be able to use their workstations effectively until the server comes back up again! Ideally, you should prevent server downtime by administering the server carefully and implementing HP's MC ServiceGuard and MirrorDisk/UX high availability solutions.

- What superuser access will be allowed? By default, the administrator of an NFS client is not allowed root access to the files stored on an NFS server. However, this security feature can be disabled on a client-by-client basis. Which clients require root access to your NFS file systems? Are the root users on those clients properly trained?

All of these questions need to be answered before you begin configuring NFS!

© 2003 Hewlett-Packard Development Company, L.P.

9–2. SLIDE: Configuring NFS Servers and Clients

Configuring NFS Servers and Clients

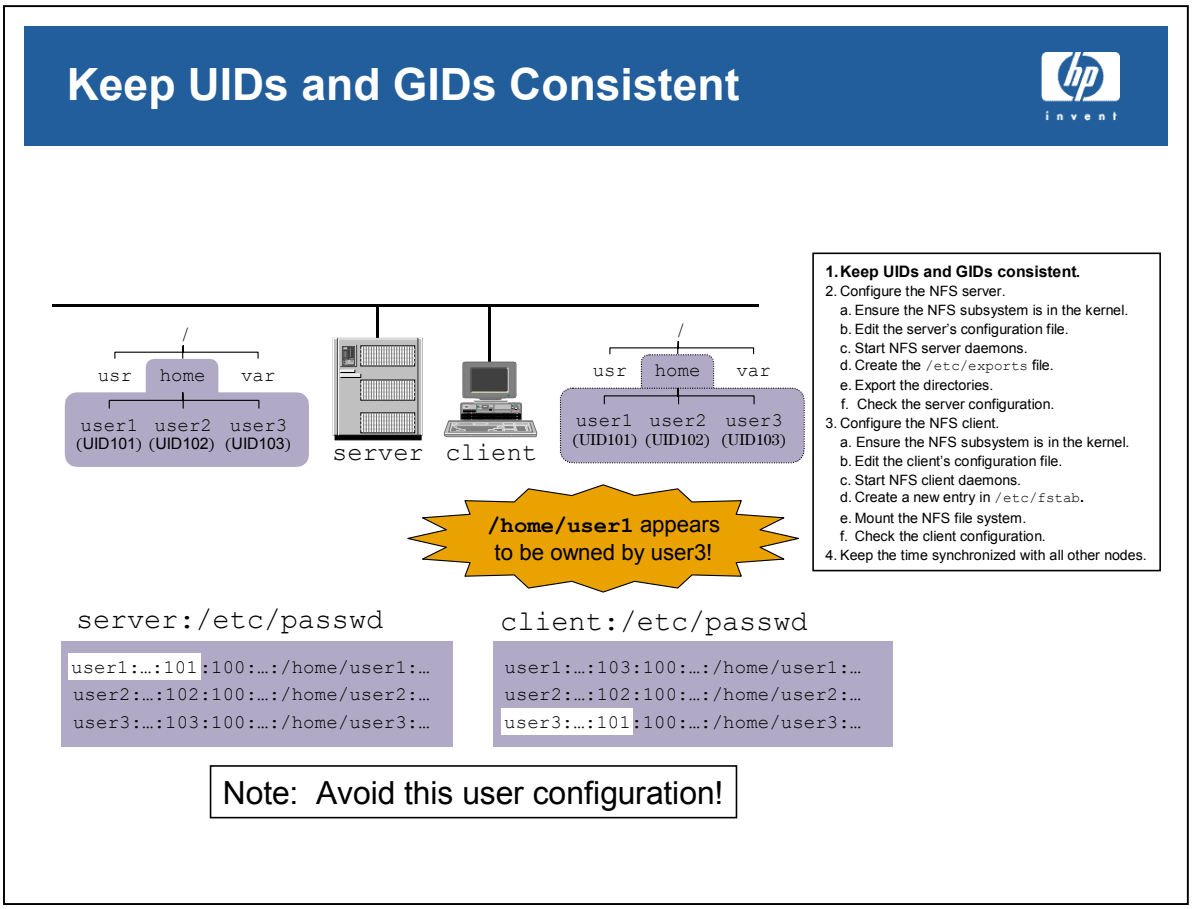
1. Keep UIDs and GIDs consistent.
2. Configure the NFS server.
   a. Ensure the NFS subsystem is in the kernel.
   b. Edit the server's `/etc/rc.config.d/nfsconf` file.
   c. Start NFS server daemons.
   d. Create the `/etc/exports` file.
   e. Export the directories.
   f. Check the server configuration.
3. Configure the NFS client.
   a. Ensure the NFS subsystem is in the kernel.
   b. Edit the client's `/etc/rc.config.d/nfsconf` file.
   c. Start NFS client daemons.
   d. Create a new entry in the `/etc/fstab` file.
   e. Mount the NFS file system.
   f. Check the client configuration.
4. Keep the time synchronized with all other nodes.

Student Notes

This slide overviews the steps that are required to configure NFS servers and clients. The remaining slides in the chapter discuss each step in detail. Note that NFS can be configured entirely via the SAM GUI/TUI interface. In order to understand better how NFS functions, the slides and notes in this course concentrate on the command-line configuration method.

ã 2003 Hewlett-Packard Development Company, L.P.

9–3. SLIDE: Keep UIDs and GIDs Consistent



Student Notes

Before you begin sharing files via NFS, it is critical to ensure that your UID and GID numbers are consistent across all the hosts in your NFS environment.

UNIX file systems identify file owners by UID number, not by username. In the example on the slide, UID 101 owns user1's home directory. UID 102 owns user2's home directory. UID 103 owns user3's home directory. These username/UID pairings are reflected in the server's **`/etc/passwd`** file.

Unfortunately, the NFS client's **`/etc/passwd`** file disagrees with the NFS server's username/UID assignments. As far as the client is concerned, all files owned by UID 101 are associated with user3, and all files owned by UID 103 are associated with user1. In this situation, it is very likely that user1 would be able to access the **`/home/user3`** home directory but not his or her own **`/home/user1`** directory.

This configuration must be avoided! Users who have logins on multiple systems must have the same UID and GID on all of those systems.

There are two ways to maintain consistent UIDs and GIDs across the network.

© 2003 Hewlett-Packard Development Company, L.P.

Maintaining UID/GID Consistency with `rcp`

In order to solve the UID/GID consistency problem, some administrators choose one host to be the keeper of the master **/etc/passwd** and **/etc/group** files and then propagate these master files to all hosts on the network on a regular basis.

A **cron** job can be scheduled on each client to automate the propagation process:

```
# vi /root/cppasswd
   #!/usr/bin/sh
   # This script is used to copy files from the master machine
   # to the localhost.
   MASTER=masterhost
   echo "Copying files from $MASTER:"
   echo group;  rcp -p $MASTER:/etc/group  /etc/group
   echo passwd; rcp -p $MASTER:/etc/passwd /etc/passwd

# chmod +x /root/cppasswd

# crontab -e
  0 1 * * * /root/cppasswd | /usr/bin/mail root
```

The script above assumes that the master server's **~root/.rhosts** file allows password free access from all other hosts on the network.

This method has several shortcomings:

- The updates occur only once per day. If a new user account is created on the master host at 2 am, the clients will not recognize the new user account until 1 am the next morning.
- All updates must be made on the master server. If a user changes his or her password on any other host on the network, the change will be overwritten the next time the script executes.
- The same root password must be used on all hosts in the NFS environment, since the root account **/etc/passwd** entry is propagated out to all the hosts every morning. Many administrators prefer to assign unique root passwords on each system to improve security.

Maintaining UID/GID Consistency with NIS or NIS+

The NFS product includes two services called "NIS" and "NIS+," which provide a much more elegant solution for maintaining UID/GID consistency. NIS will be discussed in detail in a later chapter. NIS+ is a more flexible but much more complex solution. NIS+ is discussed in HP's three-day NIS/NIS+ course (course #H3066S).

Retrofitting `/etc/passwd` and `/etc/group` for Use with NFS

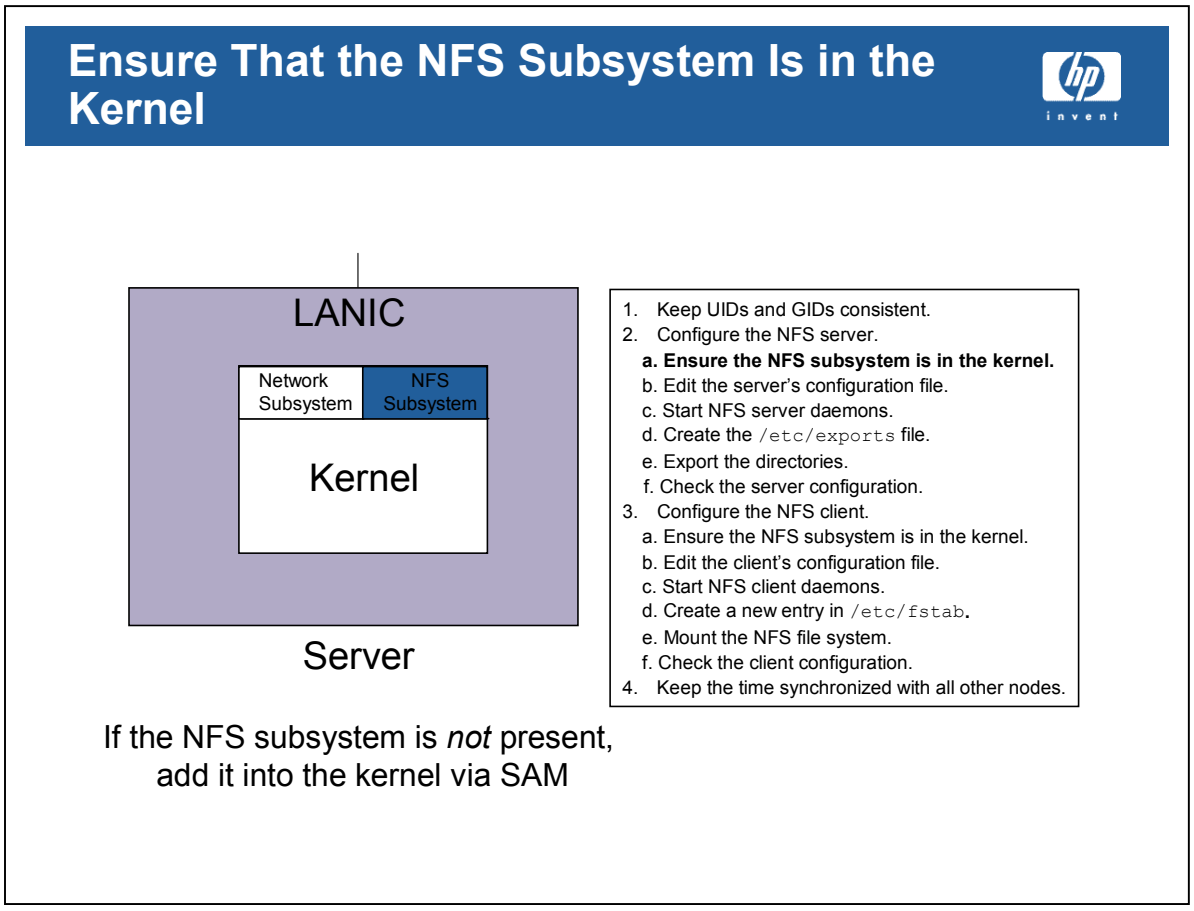
If you are installing NFS after you have been using your network for some time, you will have to modify the **/etc/passwd** and **/etc/group** files so that each user has a unique UID and a unique GID that are the same on all servers and clients.

ã 2003 Hewlett-Packard Development Company, L.P.

If you do this, your backups will become obsolete (since recovered files will have wrong ownership). Make sure you save a copy of **`/etc/passwd`** to **`passwd.old`**.

© 2003 Hewlett-Packard Development Company, L.P.

9–4. SLIDE: Ensure That the NFS Subsystem Is in the Kernel



Student Notes

The LAN/9000 (networking) subsystem and the NFS subsystem must be compiled into the server's kernel in order for NFS to work. There are several ways to verify whether the subsystems are present in the kernel.

The simplest approach is to use SAM:

```
# sam -> Kernel Configuration -> Subsystems
```

On HP-UX 10.x systems, you can use the following command:

```
# grep -n -e nfs -e lan /stand/system
```

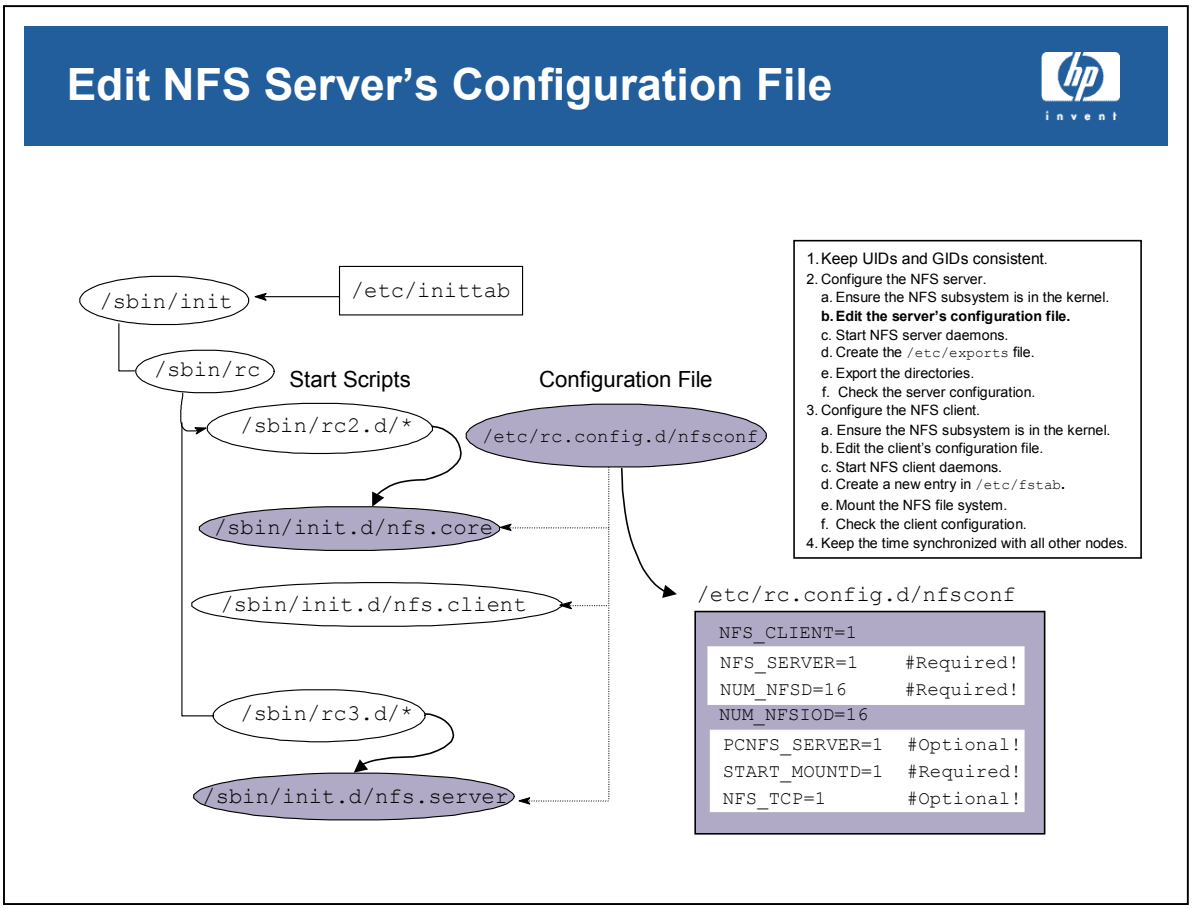
On HP-UX 11.x systems, use the following command:

```
# kmsystem | grep -e nfs -e lan
```

If either subsystem is missing, use SAM to reconfigure the kernel, then reboot.

ã 2003 Hewlett-Packard Development Company, L.P.

9–5. SLIDE: Edit NFS Server's Configuration File



Student Notes

After configuring the NFS subsystem in the kernel, you must ensure that the required NFS server daemons are started automatically during the boot process. NFS daemons, like most daemons in HP-UX, are started via startup links in the **`/sbin/rc*.d`** directories, which point to the actual startup scripts in the **`/sbin/init.d`** directory.

There are three NFS startup scripts:

`/sbin/init.d/nfs.core`	Starts the **`portmap`**/**`rpcbind`** daemons and performs other initialization tasks that are required on both NFS clients and servers. This script executes at run level 2 during system startup.
`/sbin/init.d/nfs.client`	Starts the daemons that are required on an NFS client. This script executes at run level 2.
`/sbin/init.d/nfs.server`	Starts the daemons that are required on an NFS server. This script executes at run level 3.

© 2003 Hewlett-Packard Development Company, L.P.

All three of these startup scripts share a common configuration file called **/etc/rc.config.d/nfsconf**. The NFS startup scripts read this configuration file at startup time to determine how and if NFS functionality is configured on your system. The slide above highlights the variables in **/etc/rc.config.d/nfsconf** that relate to NFS server functionality. A later slide will discuss the variables used to configure NFS client functionality.

Configuring NFS Server Variables in /etc/rc.config.d/nfsconf

Several variables in **/etc/rc.config.d/nfsconf** may need to be modified to enable and configure your NFS server appropriately.

NFS_SERVER=1 Set this variable to "1" in order to enable NFS server functionality. If this variable is set to "0," the NFS server daemons will not be started during the boot process.

NUM_NFSD=16 Every NFS client request to open, read, write or otherwise access a file or directory on an NFS file system is processed by an **nfsd** daemon running on the NFS server. Most NFS server administrators run several **nfsd** daemons in parallel to enable the server to process multiple client requests simultaneously. Generally speaking, as the number of NFS clients increases, the number of **nfsd** daemons required to service those clients will increase as well. The **NUM_NFSD** variable determines how many **nfsd** daemons should be started at boot time. In HP-UX 10.20 and standard HP-UX 11.00, the variable defaults to "4".

HP-UX 11i systems and HP-UX 11.00 systems that have the "NFS over TCP" patch installed, function a bit differently. TCP NFS requests are handled by a single, multi-threaded **nfsd** daemon. UDP NFS requests are still handled by multiple independent **nfsd** processes. On these systems that support NFS over TCP, the number of **nfsd** daemons started to handle UDP NFS requests will be set equal to the greater of either (a) four times the number of active CPUs or (b) the value of the **NUM_NFSD** variable in **/etc/rc.config.d/nfsconf**. In either case, one additional **nfsd** will be started to handle TCP NFS requests. In HP-UX 11i, the default value of the **NUM_NFSD** variable is 16, which yields 17 **nfsd**'s in the process table.

PCNFS_SERVER=1 Although NFS was originally developed to share files among UNIX systems, several vendors now offer NFS client software for the Microsoft Windows operating systems. Sharing files with Windows clients is complicated by the fact that Windows usernames and IDs are entirely different from UNIX usernames and UIDs. By default, the NFS server finesses this issue by granting all Windows clients the access rights associated with UNIX UID -2, user "nobody." Typically, this UID has very few access rights on a UNIX system.

If you wish to grant more permissive access rights to Windows clients, you must enable the **rpc.pcnfsd** server daemon by setting the **PCNFS_SERVER** variable to "1" (the default value is "0"). If the **rpc.pcnfsd** daemon is running, the server will prompt each

ã 2003 Hewlett-Packard Development Company, L.P.

Windows client for a UNIX username and password each time they mount an NFS file system. Note that **rpc.pcnfsd** is not required in order for Windows clients to mount NFS file systems; it is required only if the Windows users need to have regular user access rights to the files on the NFS server. If your server does not have any Windows clients, set **PCNFS_SERVER** default to 0.

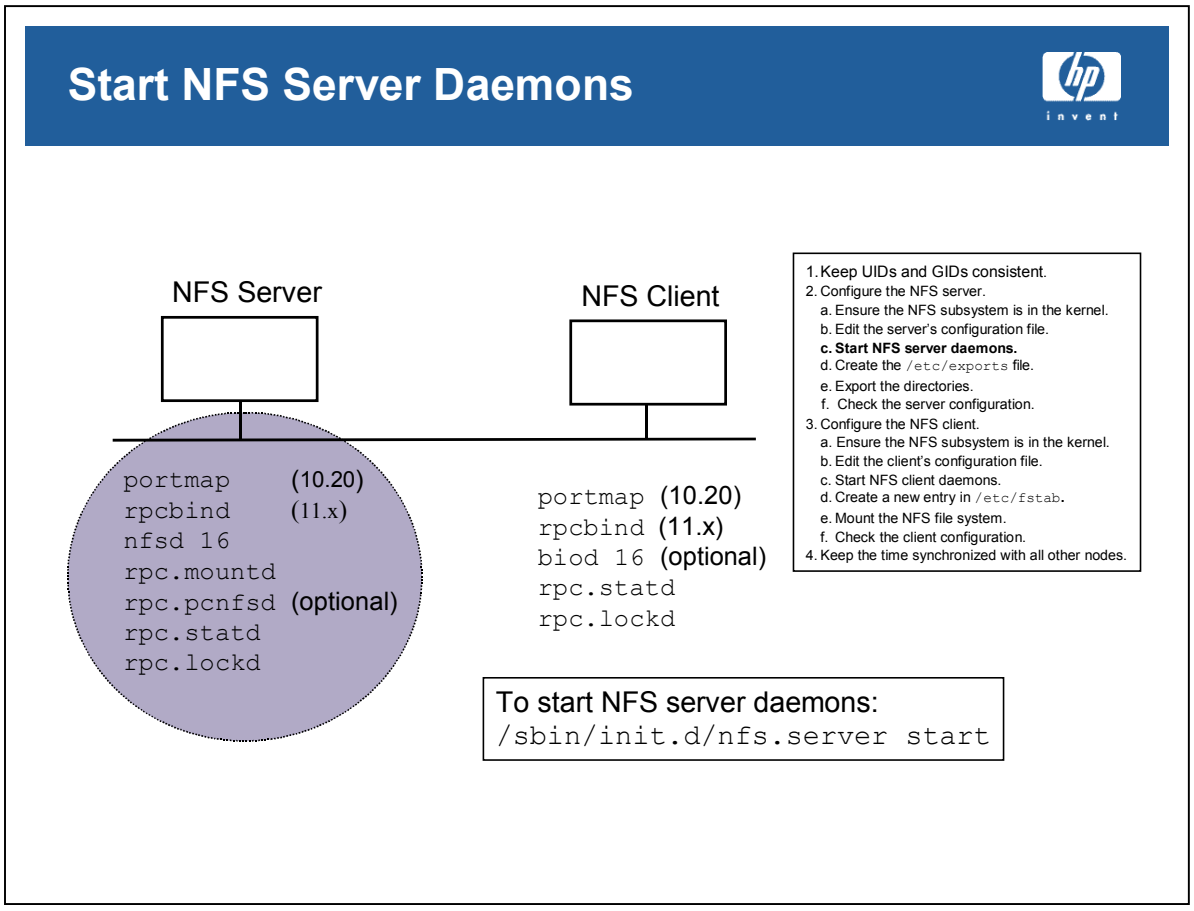
START_MOUNTD=1 This variable determines whether the **rpc.mountd** daemon should be started automatically at boot time. In HP-UX 11.x, this variable must be set to "1" on NFS servers. Before HP-UX 11.x, some administrators chose to start **rpc.mountd** via the **inetd** daemon instead; this approach is no longer supported.

NFS_TCP=1 If you are running HP-UX 11.00 and have installed the NFS over TCP patch, the TCP functionality must be enabled by setting **NFS_TCP=1** in **/etc/rc.config.d/nfsconf** (if the variable doesn't yet exist, add it to the end of the file). After making this change, both the NFS server and client daemons must be stopped and restarted. At HP-UX 11i, this variable is no longer used; NFS over TCP is enabled by default.

NOTE: If your system requires client *and* server functionality, you must configure both the server variables described here, and the client variables described later in the chapter.

© 2003 Hewlett-Packard Development Company, L.P.

9–6. SLIDE: Start NFS Server Daemons



Student Notes

After configuring the **/etc/rc.config.d/nfsconf** file as described on the previous page, you can either reboot your system or manually run the NFS server startup script to stop and restart the NFS server daemons:

```
# /sbin/init.d/nfs.server stop
# /sbin/init.d/nfs.server start
```

The startup script starts the following daemons:

portmap This daemon, used in HP-UX 10.20 and earlier releases, converts RPC program numbers into port numbers. When an RPC server program starts, it registers the following information with **portmap**:

- The port on which it is listening.
- The RPC program numbers and versions it serves.

All RPC requests from clients are initially sent to the **portmap** daemon on port number 111. **portmap** compares the "RPC Program Number" in the incoming packet against the list of registered program numbers

ã 2003 Hewlett-Packard Development Company, L.P.

to determine to which port the RPC request should be forwarded. **portmap** must be the first RPC program started and the last to die. If the **portmap** daemon dies prematurely, then it, as well as all of the registered RPC programs, must be restarted.

rpcbind This daemon is used in HP-UX 11.00 and beyond as a replacement for **portmap**.

nfsd The NFS server daemons respond to clients' file system access requests. When a client program needs to interact with a remote file system, it sends a request to one of the server's **nfsd** processes.

rpc.mountd This RPC daemon answers clients' file system mount requests. Users may also query this daemon to determine which file systems have been exported or mounted.

rpc.pcnfsd The PC server daemon is called by PC-NFS users to perform PC user authentication on HP-UX servers. This allows a PC user to access NFS file systems with the appropriate UIDs and GIDs. It also allows access to HP-UX printer facilities.

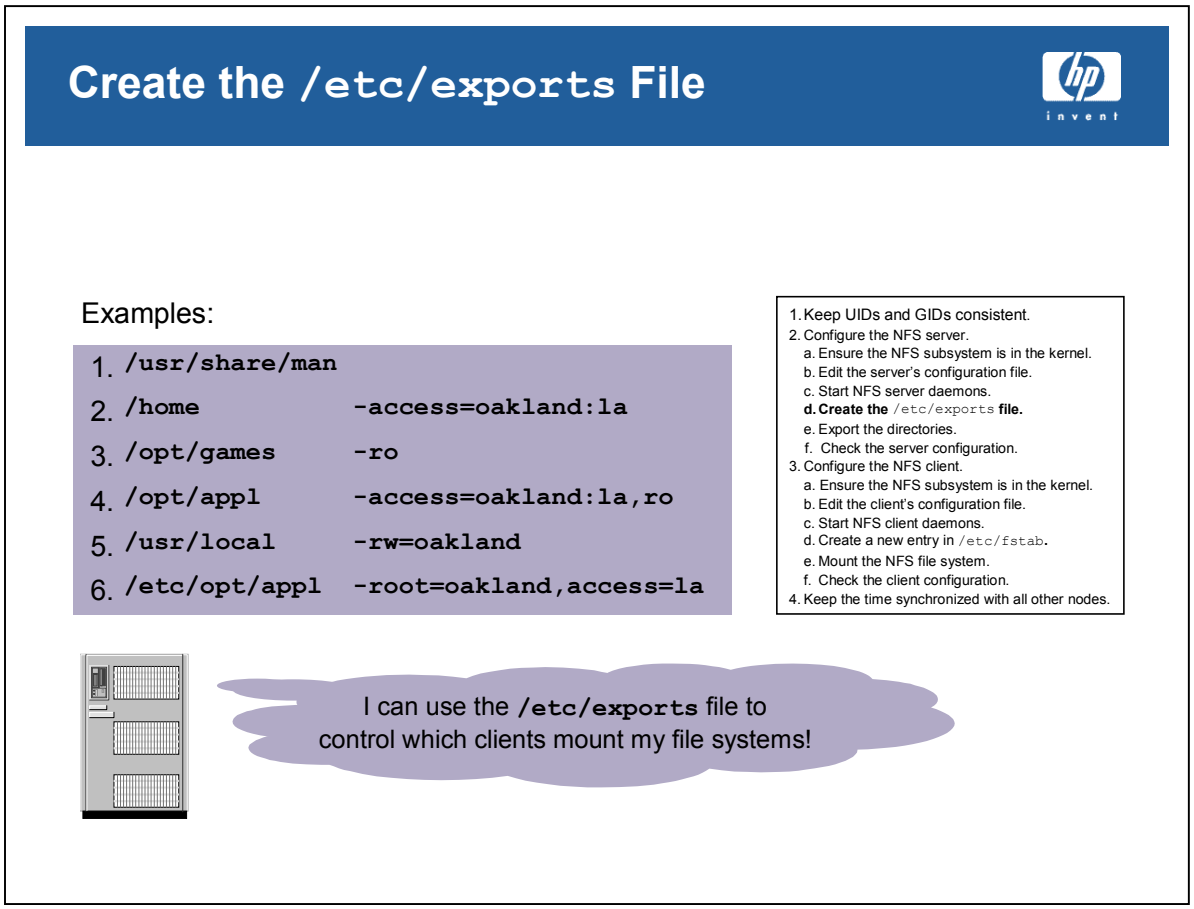
The **rpc.pcnfsd** daemon does not have to be running on the server system to use PC-NFS. If **rpc.pcnfsd** is not running, or if the PC user elects not to log in to the server system, the PC user becomes **nobody** on the server system with the permissions of **other**.

rpc.lockd When an application is processing a critical file, the application may place a "lock" on the file to prevent other processes from modifying the file for a period of time. The NFS server's **rpc.lockd** daemon listens for lock requests from NFS clients and locks the requested files accordingly. However, locks requested via **rpc.lockd** are not really enforced. **rpc.lockd** simply creates a flag, or "semaphore," indicating that a process has requested a lock on the file. Other processes may choose to honor or ignore the lock flag. See the **rpc.lockd(1m)** and **lockf(2)** man pages for details.

rpc.statd When an NFS client places a lock on a file via **rpc.lockd**, the server's **rpc.statd** daemon is responsible for periodically verifying that the client is still functioning. If the client reboots unexpectedly, **rpc.statd** automatically removes all locks placed by the client to allow other processes to again access the client's locked files.

© 2003 Hewlett-Packard Development Company, L.P.

9–7. SLIDE: Create the `/etc/exports` File



Student Notes

After starting the NFS server daemons, you must configure the **`/etc/exports`** file to specify which file systems you want to share with your NFS clients.

Each line in the **`/etc/exports`** file has two fields. The first field identifies a file system, directory, or file that should be made available to NFS clients. NFS provides a great deal of flexibility. If the first field identifies a directory that serves as a mount point for a local file system, that entire file system is made available to clients. If you only want to share a subdirectory tree within a file system, specify that subdirectory path in the first field. In fact, you can even export a single file!

The second field determines which clients can mount the file system and what those clients are allowed to do in the file system. Clients that are granted "read-only" access can view the files and directories in the file system, but cannot make changes. Clients that are granted "read-write" access can both view and modify the files and directories in the file system. Note that the options in **`/etc/exports`** never mention "execute" permission. As far as the export options are concerned, clients that have "read" access should be allowed to read executable code into memory and execute it.

ã 2003 Hewlett-Packard Development Company, L.P.

The export options supplement, but do not replace normal UNIX file permissions. If the permissions on a file are set to "000", none of the clients will be allowed to view, modify, or execute the file regardless of the export options specified in **/etc/exports.**

The table below shows the most common export option combinations. The first column shows several common combinations of export options. The remaining three columns indicate which clients would be able to access each file system, and how, given the access option listed on the left (rw="read and write access allowed", ro="read-only access allowed"). Look at the table, then see if you can guess which clients will be able to mount each file system on the slide. (The slide examples are explained at the end of the notes accompanying this slide.)

Table 1

export options used:	**hosta**	**hostb**	**others**
/home –access=hosta	rw	—	—
/home –access=hosta:hostb	rw	rw	—
/home	rw	rw	rw
/home –rw=hosta:hostb	rw	rw	ro
/home –rw=hosta	rw	ro	ro
/home –ro	ro	ro	ro
/home –access=hosta:hostb,ro	ro	ro	—
/home –access=hosta,ro	ro	—	—

By default, **root** on the client systems is treated as user **nobody** when processing files on NFS servers. In order to grant NFS clients **root** access, the **root** option to the **export** command must be used. If a file system is exported to a client with the **root** option, then the user **root** on that client will have **root** permission on that file system. The table below shows several examples using the **root** export option:

Table 2

export options used:	**hosta**	**hostb**	**others**
/home –root=hosta,access=hosta	root+rw	—	—
/home –root=hosta,access=hosta:hostb	root+rw	rw	—
/home –root=hosta	root+rw	rw	rw
/home –root=hosta,rw=hosta:hostb	root+rw	rw	ro
/home –root=hosta,rw=hosta	root+rw	ro	ro
/home -root=hosta,rw=hosta,access=hosta:hostb	root+rw	ro	—
/home –root=hosta,access=hosta	root+rw	—	—

© 2003 Hewlett-Packard Development Company, L.P.

Syntax of /etc/exports

A more formal description of the **/etc/exports** follows below. Export options in **/etc/exports** are preceded with a dash, and are separated by commas. Some export options require a list of hostnames as arguments. Hostnames in these lists must be separated by colons.

The export options are as follows:

ro	Exports the directory read-only. This prevents hosts from writing to the file system.
rw=*hostname*[:*hostname*]	Exports the directory "read-mostly." This limits read-write capability to the specified hosts. Clients that are not explicitly listed after the **rw=** can still mount the file system, but will not be allowed to make changes. Up to 256 host names can be specified.
anon=*uid*	If an NFS request comes from an unknown user, grant that user the privileges normally associated with ***uid***. Remote root users (UID 0) are always treated as anonymous users by the NFS server unless their username is included in the **-root=** export list. If **rpc.pcnfsd** is disabled, then users on Windows clients will also be treated as "unknown" users, too. An unknown user has the UID -2 by default, which maps to username **nobody** in the **/etc/passwd** file. `nobody:*:-2:-2::/:`
root=*hostname*[:*hostname*]	Gives root (superuser) access only to root users from a specified host name or hosts. By default, no hosts are granted root access. Up to 256 hostnames can be specified.
access=*client*[:*client*]	Allows mount access to the specified client or clients. A client can either be a host name or a netgroup. Each client in the list is first checked in the **/etc/netgroup** database.
async	Increases the write performance on the NFS server by causing asynchronous writes on the NFS server. The **async** option can be specified anywhere on the command line after the directory name.

ã 2003 Hewlett-Packard Development Company, L.P.

Explanation of the Examples

Were you able to guess which clients could mount each file system on the slide? Read the explanations below if you need help.

1. **`/usr/share/man`**

   Exports the **`man`** pages with read-write access to every client.

2. **`/home -access=oakland:la`**

   Exports **`/home`** with read-write access for oakland and la. Other hosts will not be allowed to mount the file system at all.

3. **`/opt/games -ro`**

   Exports the games directory with read-only access for all hosts.

4. **`/opt/appl -access=oakland:la,ro`**

   Exports with read-only access for oakland and la. No other clients will be allowed to mount the file system.

5. **`/usr/local -rw=oakland`**

   Exports with read-write access for oakland, and read-only access for all other hosts.

6. **`/etc/opt/appl -root=oakland,access=la`**

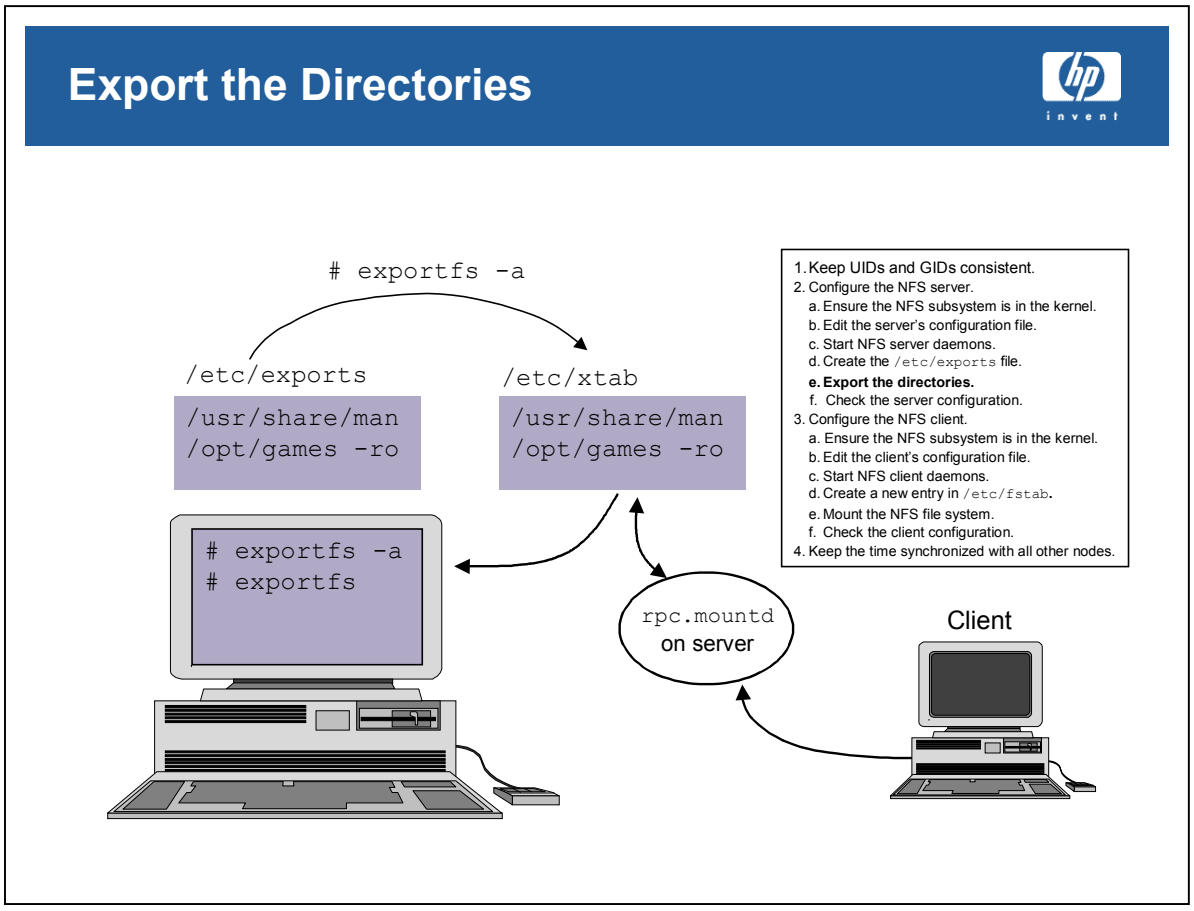
   Grants **`root`** on oakland UID 0 access to the file system. Also allows read-write access for host la. Other hosts will not be allowed to mount the file system at all.

CAUTION:	Export directories and file systems on an as-needed basis only. Always use export options to restrict access rights.

NOTE:	You cannot export either a parent directory or a subdirectory of an exported directory that resides within the same file system. It is not possible, for instance, to export both **`/usr`** and **`/usr/local`**, if both directories reside in the same file system.

© 2003 Hewlett-Packard Development Company, L.P.

9–8. SLIDE: Export the Directories



Student Notes

Simply adding a file system or directory to **/etc/exports** does not immediately make that file system available to clients. Any time the **/etc/exports** file is modified, the administrator must notify the **rpc.mountd** daemon that a change has occurred by executing the **exportfs** command:

```
# exportfs -a
```

The **/sbin/init.d/nfs.server** script executes this command automatically at boot time to initially export all file systems.

Several other options on **exportfs** are also available:

`# exportfs`	Lists all currently exported file systems.
`# exportfs -i /home`	Exports a file system without adding it to **/etc/exports**.
`# exportfs -u /home`	Unexports a file system.
`# exportfs -a`	Exports all file systems listed in **/etc/exports**.
`# exportfs -ua`	Unexports all file systems listed in **/etc/exports**.

ã 2003 Hewlett-Packard Development Company, L.P.

The superuser can execute the **exportfs** command at any time to alter the list or characteristics of exported directories. It must be invoked every time **/etc/exports** is modified.

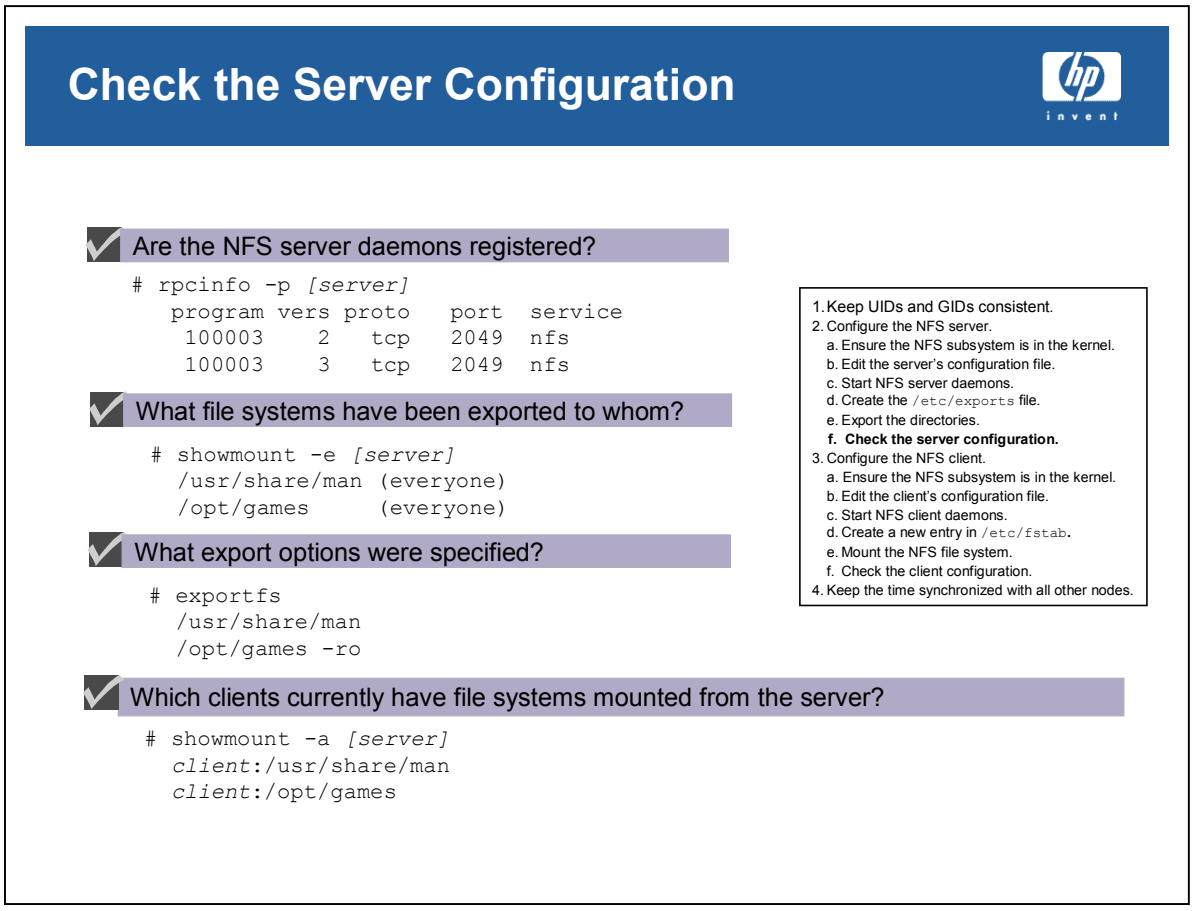
If an NFS mounted directory is unexported via **exportfs -u**, clients that have already mounted the file system will receive NFS file handle errors when they attempt to access the unexported file systems. The client administrators can remove the "stale" file system from the mount table via the **umount** command.

Internally, the **exportfs** command functions by simply adding and removing entries from a file called **/etc/xtab** which the **rpc.mountd** daemon uses to determine which file systems have been made available to which clients. Exporting a file system adds a line to **/etc/xtab**, and unexporting a file system disables or removes an entry from the **/etc/xtab** file. Executing **exportfs** without any options simply displays the contents of the **/etc/xtab** file.

NOTE: The server must have the directory locally mounted before it can be exported.

© 2003 Hewlett-Packard Development Company, L.P.

9–9. SLIDE: Check the Server Configuration



Student Notes

After completing the NFS server configuration, check your work.

Are the NFS server daemons registered?

First, verify that the NFS daemons started properly and registered themselves with the **rpcbind/portmap** daemon. Use the **rpcinfo -p** command to query your server's **portmap/rpcbind** daemon for a list of registered RPC programs.

```
# rpcinfo -p [servername]
```

At a minimum, make sure that you see **mountd** and **nfs** in the resulting list. If either of these programs is missing, you may need to restart the NFS server functionality:

```
# /sbin/init.d/nfs.server stop
# /sbin/init.d/nfs.server start
```

Look in the second column of the output to determine which versions are supported. Does your server's **nfs** program support NFS PV3? The third column indicates which transport protocol(s) your **nfs** daemon supports. Does your system support NFS over TCP?

ã 2003 Hewlett-Packard Development Company, L.P.

What file systems have been exported to whom?

Next, determine which clients have access to your exported file systems. The **showmount -e** command queries your **rpc.mountd** daemon to obtain this information:

```
# showmount -e
```

The command should list all exported file systems, and the clients that have access to each file system. If file systems or clients are missing, you may need to re-execute the **exportfs** command.

What export options were specified?

Although **showmount** lists the exported file systems, it does not indicate which clients get read, write, and root access. Execute the **exportfs** command to verify your export options:

```
# exportfs
```

Which clients currently have file systems mounted from the server?

If you want to determine which clients are actually using your NFS file systems, execute the **showmount -a** command:

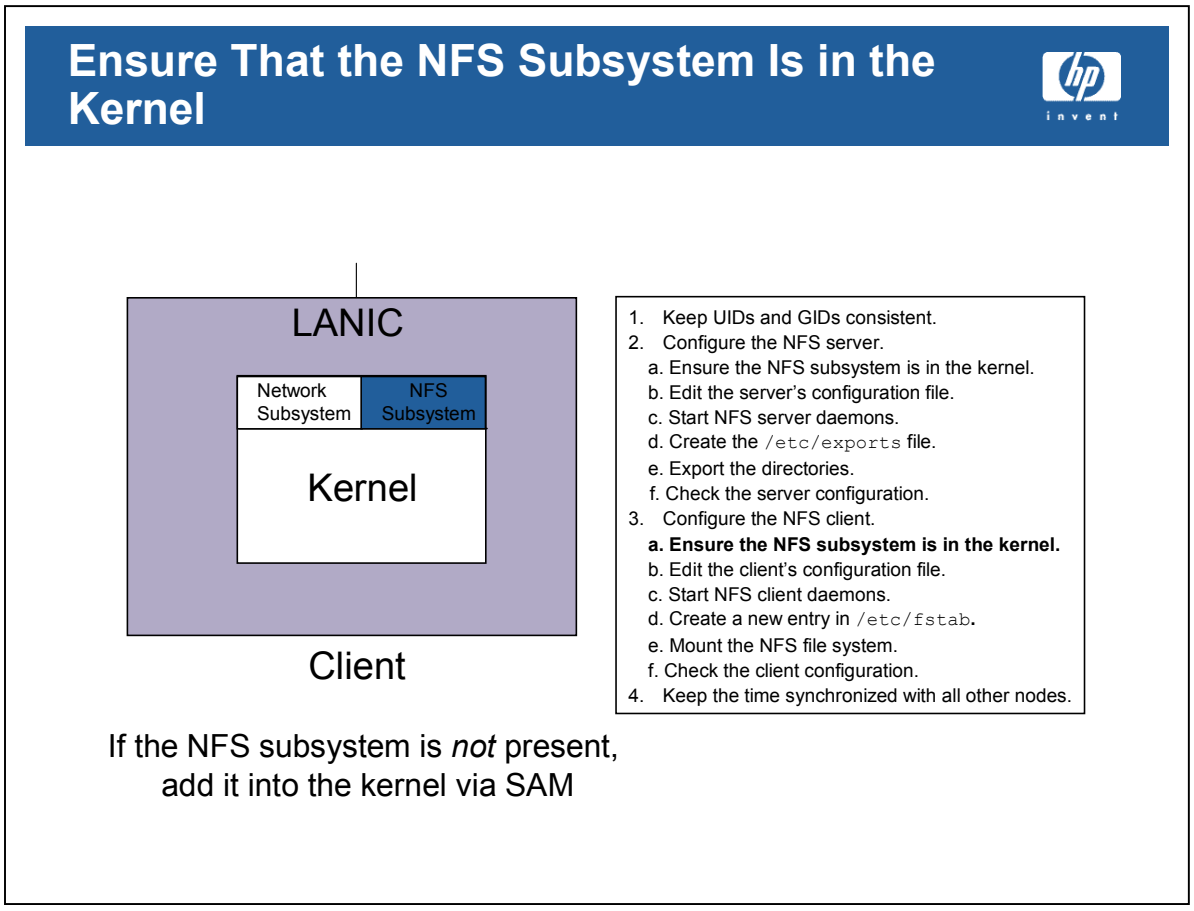
```
# showmount -a
```

This command displays the contents of the **/etc/rmtab** (remote mount table) file in a human-readable format. Every time a client mounts a file system, the **rpc.mountd** daemon adds a line to the remote mount table in **/etc/rmtab**. Theoretically, the **rpc.mountd** daemon then removes clients from **rmtab** as file systems are unmounted. However, if a client crashes or loses connectivity to the NFS server, **showmount -a** may list clients that no longer have your file systems mounted. You can purge all entries from the **/etc/rmtab** file by executing:

```
# > /etc/rmtab
```


© 2003 Hewlett-Packard Development Company, L.P.

9–10. SLIDE: Ensure That the NFS Subsystem Is in the Kernel



Student Notes

NFS clients, like NFS servers, must have the LAN and NFS subsystems configured in the kernel.

SAM provides the simplest mechanism for viewing and modifying the kernel:

```
# sam -> Kernel Configuration -> Subsystems
```

On HP-UX 10.x systems, you can use the following command to view the contents of the kernel:

```
# grep -n -e nfs -e lan /stand/system
```

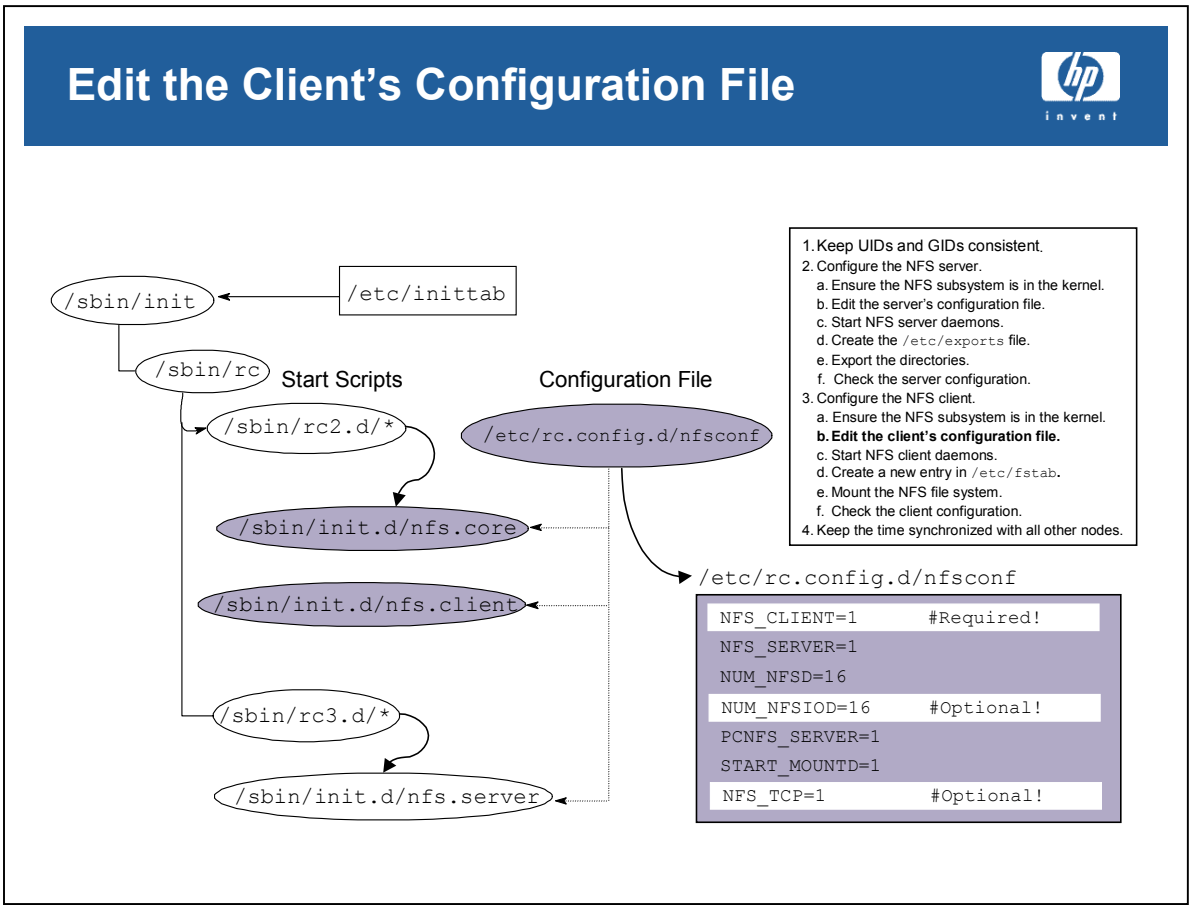
On HP-UX 11.x systems, use the following command:

```
# kmsystem | grep -e nfs -e lan
```

```
If either subsystem is missing, use SAM to reconfigure the kernel,
then reboot.
```


ã 2003 Hewlett-Packard Development Company, L.P.

9–11. SLIDE: Edit the Client's Configuration File



Student Notes

After configuring NFS client functionality in the kernel, there are several variables in the **/etc/rc.config.d/nfsconf** file that may need to be modified to enable and configure your NFS client:

NFS_CLIENT=1 Set this variable to "1" to ensure that **/sbin/init.d/nfs.client** executes during system startup.

NUM_NFSIOD=16 This variable determines the number of **/usr/sbin/biod** (Block I/O Daemons) that are started during the boot process. **biod** daemons enable NFS to provide buffer cache read-ahead and write-behind access to NFS file systems. This number may need to be increased on clients that use NFS heavily. Up through HP-UX 11.00, **NUM_NFSIOD** defaults to "4". At 11i, the default value is "16".

NFS_TCP=1 If you are running HP-UX 11.00 and have installed the NFS over TCP patch, the TCP functionality must be enabled by setting **NFS_TCP=1** in **/etc/rc.config.d/nfsconf** (if the variable doesn't yet exist, add it to the end of the file). After making this change, both the NFS server

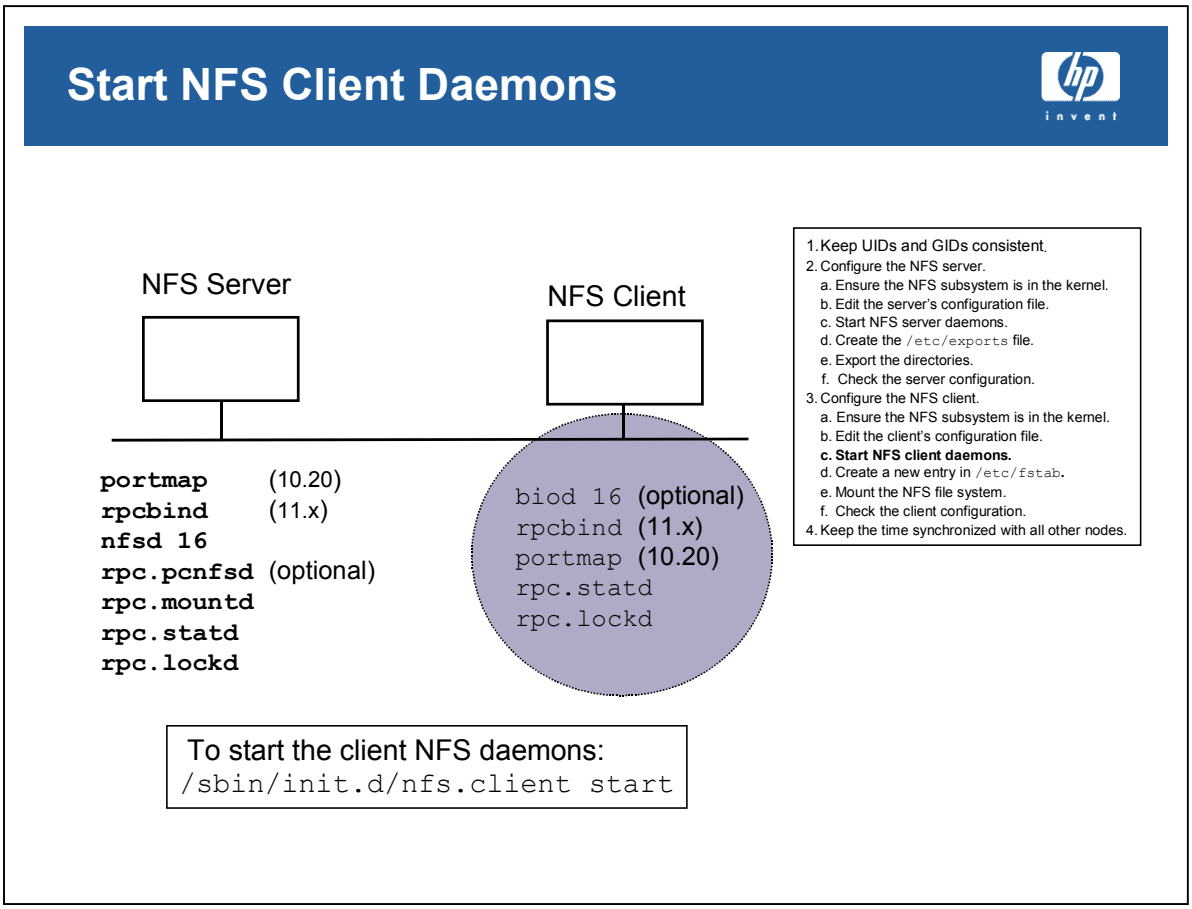
© 2003 Hewlett-Packard Development Company, L.P.

and client daemons must be stopped and restarted. At HP-UX 11i, this variable is no longer used; NFS over TCP is enabled by default.

NOTE:	If your system requires client *and* server functionality, you must configure both the client variables listed here and the server variables described earlier in the chapter.

ã 2003 Hewlett-Packard Development Company, L.P.

9–12. SLIDE: Start NFS Client Daemons



Student Notes

After modifying the **/etc/rc.config.d/nfsconf** file, you can either reboot or manually execute the NFS client startup script to stop and restart the NFS client daemons:

```
# /sbin/init.d/nfs.client stop
# /sbin/init.d/nfs.client start
```

The startup script starts the following daemons:

portmap This daemon, used in HP-UX 10.20 and earlier releases, converts RPC program numbers into port numbers. When an RPC server program starts, it registers the following information with **portmap**:

- The port on which it is listening.
- The RPC program numbers and versions it serves.

All RPC requests from clients are initially sent to the **portmap** daemon on port number 111. **portmap** compares the "RPC Program Number" in the incoming packet against the list of registered program numbers to determine which port the RPC request should be forwarded to.

© 2003 Hewlett-Packard Development Company, L.P.

portmap must be the first RPC program started, and the last to die. If the **portmap** daemon dies at any point, then it, as well as all of the registered RPC programs, must be restarted.

rpcbind This daemon is used in HP-UX 11.00 and beyond as a replacement for **portmap**.

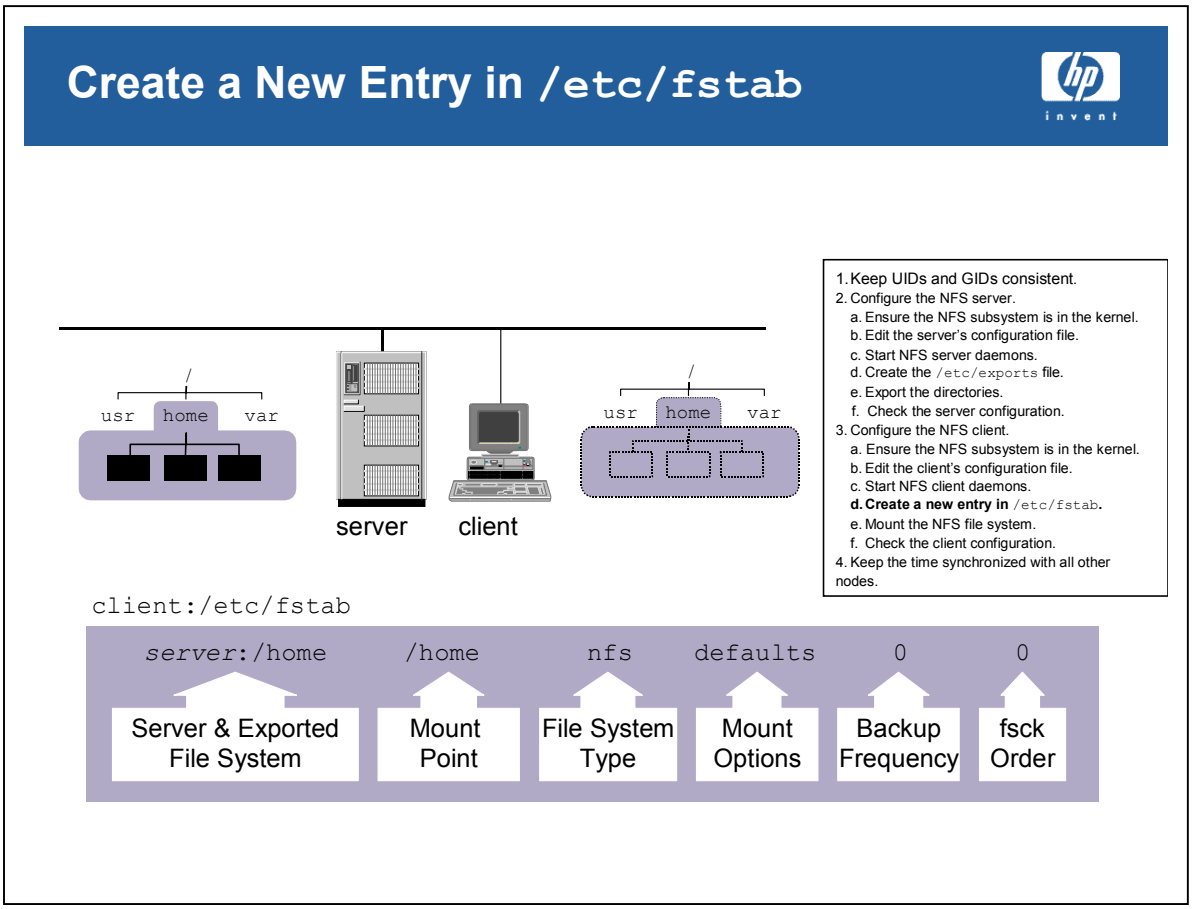
biod The asynchronous block I/O daemons are used by NFS clients to handle buffer cache read-ahead and write-behind.

rpc.lockd When an application is processing a critical file, the application may place a "lock" on the file to prevent other processes from modifying the file for a period of time. NFS clients use the **rpc.lockd** daemon to request locks on files in the NFS file system. However, locks requested via **rpc.lockd** are not really enforced. **rpc.lockd** simply creates a flag, or "semaphore," indicating that a process has requested a lock on the file. Other processes may choose to honor or ignore the lock flag. See the **rpc.lockd(1m)** and **lockf(2)** man pages for details.

rpc.statd When an NFS client places a lock on a file via **rpc.lockd**, the server's **rpc.statd** daemon is responsible for periodically verifying that the client is still functioning by periodically attempting to contact the client's **rpc.statd** daemon. If the client reboots unexpectedly, the server's **rpc.statd** daemon automatically removes all locks placed by the client to allow other processes to again access the client's locked files.

ã 2003 Hewlett-Packard Development Company, L.P.

9–13. SLIDE: Create a New Entry in `/etc/fstab`



Student Notes

After enabling NFS client functionality, you must specify which NFS file systems you wish to mount. You can manually mount and unmount NFS file systems via the **`mount`** and **`umount`** commands, or you can ensure that your NFS file systems mount automatically at boot time by adding them to the **`/etc/fstab`** file. This slide concentrates on **`/etc/fstab`**; the next slide details some of the options available on the **`mount`** and **`umount`** commands.

NFS **`/etc/fstab`** entries are very similar to VxFS and HFS entries in the **`/etc/fstab`** file:

Server and Exported FS: Identifies the NFS server hostname and the pathname on the server for the file system you wish to mount. The hostname must be separated from the pathname by a colon (:).

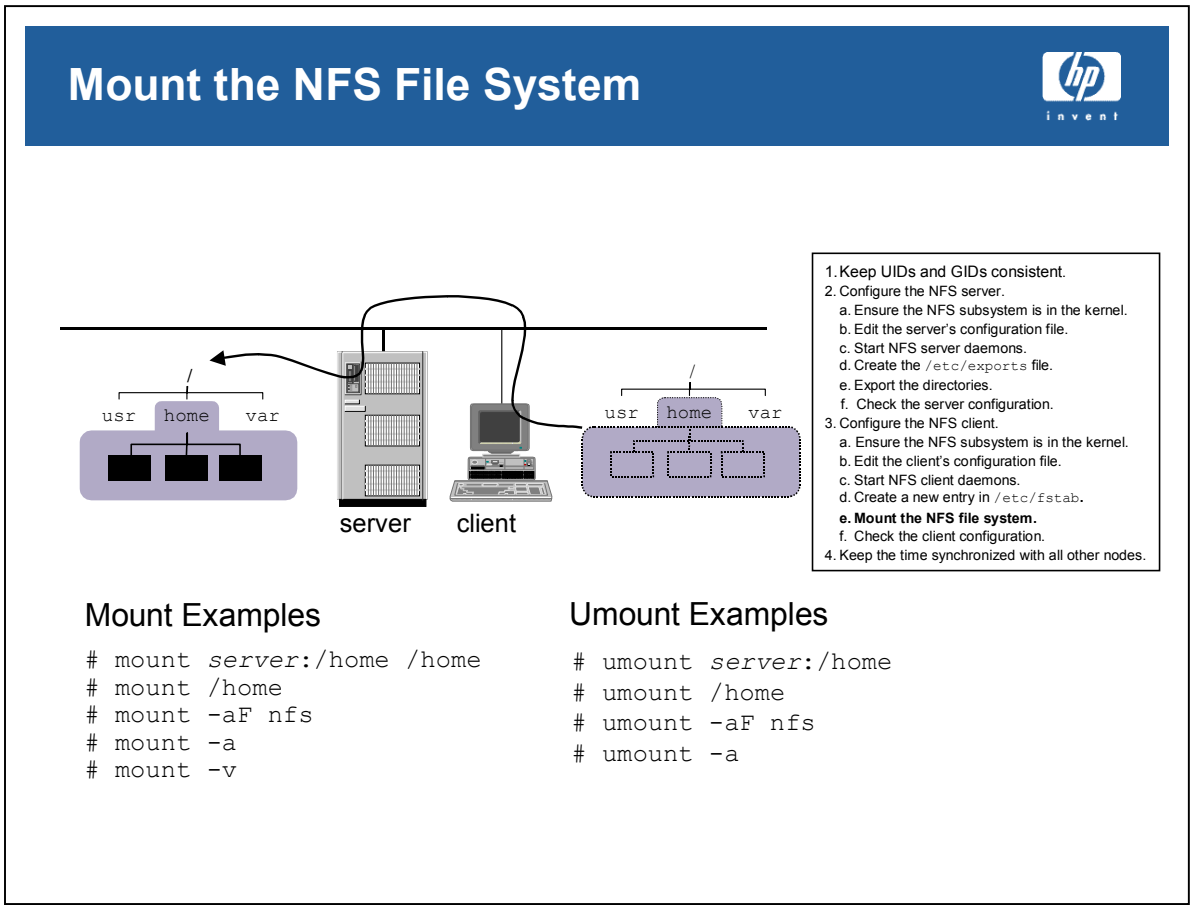
If you wish, you can mount a portion of an exported file system rather than the entire exported file system. For instance, if the NFS server exported the **`/home`** file system, you could mount **`/home`** and everything under it, or you could choose a single subdirectory to mount (for example, **`/home/user1`**).

© 2003 Hewlett-Packard Development Company, L.P.

	Whatever you choose to mount, be sure to identify the file system you choose via a full pathname!
Mount Point:	Identifies the mount point that should be used on the NFS client. The client's mount point need not match the pathname used on the NFS server side. If any local files reside under the specified mount point directory, the local files will be hidden as long as the NFS file system is mounted. Ideally, the mount point directory should be an empty directory. Be sure to use a full pathname when specifying the mount point directory!
File System Type:	Set to **`nfs`** for NFS file systems. During the system startup process, the **`/sbin/init.d/nfs.client`** startup script mounts all **`nfs`** type file systems that are listed in **`/etc/fstab`**. Other startup scripts also use the **`fstab`** file, too: **`/sbin/init.d/localmount`** mounts all **`hfs`** and **`vxfs`** file system entries, and **`/sbin/init.d/swap_start`** enables all of the **`swap`** type entries.
Mount Options:	The **`mount`** command recognizes a variety of mount options that determine how a file system may be accessed. The notes accompanying the next slide describe NFS mount options in detail. If you simply want to accept the default options, use the keyword **`defaults`** in this field.
Backup Frequency:	This field is unused currently in HP-UX, but requires a "0" placeholder.
`fsck` Order:	After an improper system shutdown, HP-UX automatically executes the **`fsck`** command to identify and fix file system corruption. The "fsck Order" field determines the order in which **`fsck`** checks your file systems. Since **`fsck`** can only be executed on local file systems, this field should be set to "0" for NFS entries in **`/etc/fstab`**.

ã 2003 Hewlett-Packard Development Company, L.P.

9–14. SLIDE: Mount the NFS File System



Student Notes

The same **mount** and **umount** commands that you have used in the past to mount and unmount local file systems can also be used to mount and unmount NFS file systems.

Mount Examples

The slide shows the most common permutations of the **mount** command:

1. **mount *server*:/home /home**

   Mounts **/home** from the designated server.

2. **mount /home**

   Mounts **/home** using the associated entry in the **/etc/fstab** file.

3. **mount -aF nfs**

   Mounts all NFS type file systems that are listed in the **/etc/fstab** file.

© 2003 Hewlett-Packard Development Company, L.P.

4. **`mount -a`**

   Mounts all file systems listed in the **`/etc/fstab`** file.

5. **`mount -v`**

   Lists all file systems that are currently mounted.

Umount Examples

In order to unmount NFS file systems, use the **`umount`** command. The **`umount`** command recognizes several options and arguments:

1. **`umount server:/home`**

   Unmounts the specified NFS file system.

2. **`umount /home`**

   Unmounts the NFS file system mounted under the directory **`/home`**.

3. **`umount -aF nfs`**

   Unmounts all currently mounted NFS file systems.

4. **`umount -a`**

   Unmounts all file systems, including NFS and locally mounted file systems.

The examples on the slide show the most common mount options and arguments, but NFS also supports several other options. Some of the other NFS mount options are summarized in the remaining sections below.

Mount Options Common to All File System Types

The options described in this section apply to all file system types, including NFS.

`rw/ro`	Allow/deny users on this client the ability to make changes on the NFS file system. The default is **`rw`**.
`suid/nosuid`	Enable/disable "Set User ID" execution functionality in the NFS file system. SUID functionality makes it possible for regular users to gain temporary root privileges when executing programs that have the SUID bit set. SUID executables have been known to cause security problems in the past, so many NFS administrators choose to disable this functionality wherever possible by mounting NFS file systems **`nosuid`**. The default is **`suid`**.
`quota/noquota`	Enable/disable quota checking. See the **`quota(5)`** man page for more information. The default is **`quota`**.

ã 2003 Hewlett-Packard Development Company, L.P.

Mount Options Associated with NFS Stability and Recovery Issues

A non-responsive NFS server can cause severe problems for NFS clients. Several mount options can be used to mitigate the effect that a downed server has on its clients.

There are two very distinct issues to consider when an NFS server crashes or loses connectivity to its clients: (1) What happens to new clients that attempt to mount from the downed server? (2) What happens to existing clients that attempt to access files and directories in an already mounted file system? The table below summarizes the mount options that determine the answers to these questions. Note that some mount options affect mount request behavior, while others affect file access attempt behavior.

Mount Options Used	Mount Requests	Access Requests
`fg,retry=5`	Retry failed mount attempts 5 times before quitting. The **`mount`** command hangs until either (1) the file system successfully mounts, or (2) all 5 mount attempts timeout, which may take several minutes.	N/A
`bg,retry=1000`	Initially attempts the mount request in the foreground. If that attempt fails, retry the mount 1000 times in the background, and allow the user to proceed on to other tasks in the meantime.	N/A
`hard,intr`	N/A	Access requests hang indefinitely until the server responds. However, users may interrupt hung access requests by hitting ^C.
`hard,nointr`	N/A	Access requests hang indefinitely until the server responds. Users may not interrupt hung access attempts.
`soft,retrans=5`	N/A	Access attempts are retransmitted 5 times. After 5 failed attempts, the access request fails.

By default, NFS file systems are mounted with the **`fg,retry=1,hard,intr`** options from the table above.

Mount Options Associated with NFS PV3 Functionality

`vers=3/2` Determines whether the file system is mounted using NFS PV3 or NFS PV2. NFS PV3 made it possible to access "large files" over 2 GB in size and introduced some performance enhancements. PV2 was the only protocol version supported prior to HP-UX 11.00. When PV3 was

© 2003 Hewlett-Packard Development Company, L.P.

released with 11.00, it was backported to HP-UX 10.20 as a patch.

If the client supports NFS PV3, it will attempt to mount file systems using the PV3 protocol. If a queried server does not support PV3, the client mounts using NFS PV2. Most administrators allow the client and server to automatically negotiate a mutually acceptable protocol version. However, you may force a file system to mount using PV2 by specifying the **vers=2** mount option if you know that your server does not support PV3.

proto=tcp/udp

When NFS was originally released for HP-UX, it used the UDP protocol and was supported only on local area networks, not WANs. HP-UX 11i introduced support for NFS over TCP to enable WAN access to NFS file systems. This functionality has been backported by patch to HP-UX 11.00.

You can determine if your NFS file systems are mounted using NFS over TCP by executing the **netstat -a | grep nfs** command. If your file systems are mounted via NFS over TCP, you should see an **ESTABLISHED** TCP connection between the client and server.

By default, if NFS over TCP is enabled on a client, the client will attempt to mount all NFS file systems via TCP. If the queried server does not support NFS over TCP, the client automatically reverts to NFS over UDP.

You can force the client to use UDP by including the **proto=udp** mount option. On a local area network, UDP may be slightly more efficient, but most administrators simply accept the default TCP behavior on clients that support NFS over TCP.

Default Mount Options

If you mount a file system without specifying any mount options, or if you use the **defaults** entry in **/etc/fstab**, you get the following defaults at HP-UX 11i (the **vers=** and **proto=** options used depend on the NFS version running on the client and server):

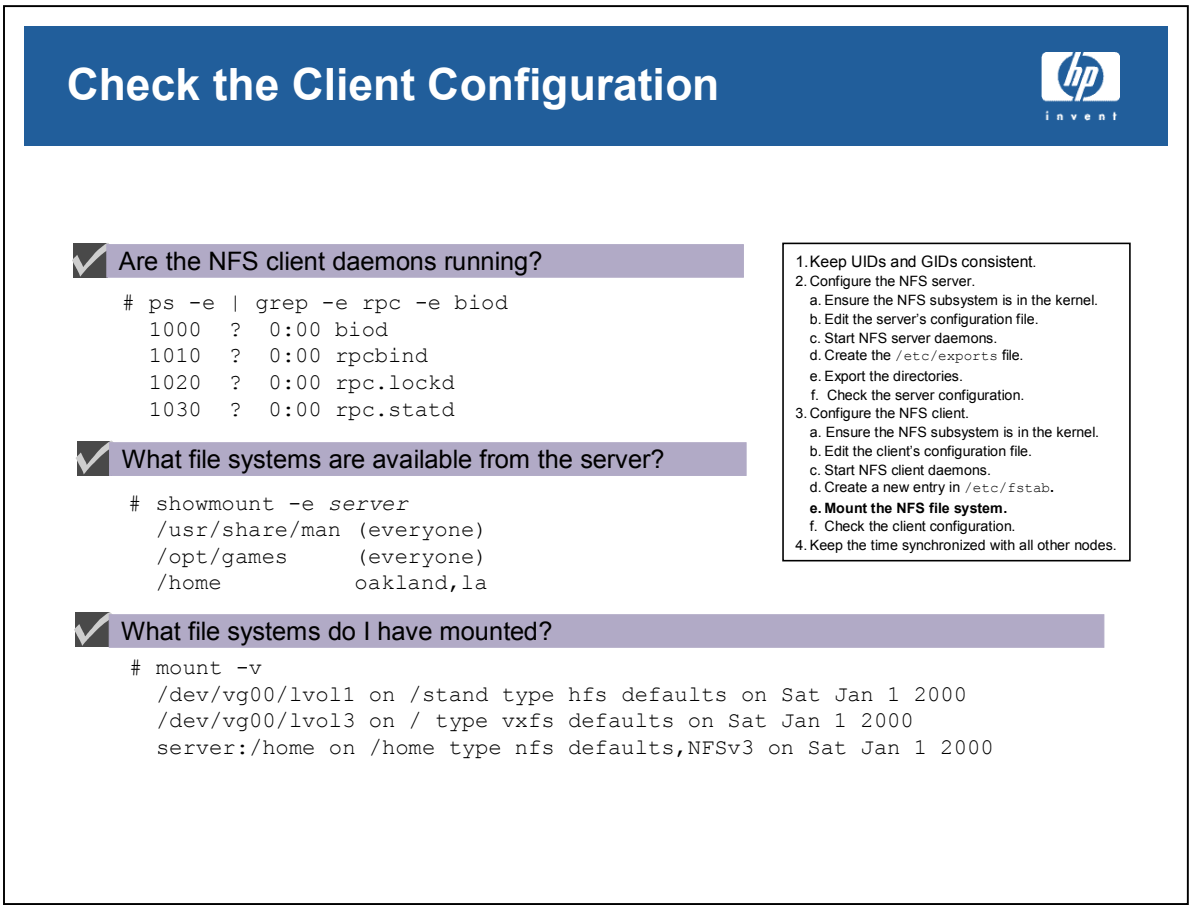
```
rw,suid,quota,fg,retry=1,hard,intr
```

Thus, the following three commands all have the same effect (assuming the **/etc/fstab** file uses the **defaults** mount option):

```
# mount svrname:/xxxx /xxxx
# mount -o defaults svrname:/xxxx /xxxx
# mount -o rw,suid,quota,fg,retry=1,hard,intr  svrname:/xxxx /xxxx
```


ã 2003 Hewlett-Packard Development Company, L.P.

9–15. SLIDE: Check the Client Configuration



Student Notes

Several commands are available for checking your NFS client configuration.

Are the NFS client daemons running?

Several daemons should be running on an NFS client. Use the **ps** command to view the process table, and look for **portmap/rpcbind**, **rpc.lockd**, and **rpc.statd**:

```
# ps -e | grep -e rpc -e biod
```

If you set the **NUM_NFSIOD** variable to a value greater than zero, you should also see several **biod** daemons running, too.

What file systems are available from the server?

Next, check to see which file systems your NFS server has made available to you by executing the **showmount -e** command:

```
# showmount -e server
```


© 2003 Hewlett-Packard Development Company, L.P.

What file systems do I have mounted?

Finally, verify that all the file systems that you added to your **/etc/fstab** file are mounted:

```
# mount -v
```


ã 2003 Hewlett-Packard Development Company, L.P.

9–16. SLIDE: Review: Configuring NFS Servers and Clients

Review: Configuring NFS Servers and Clients

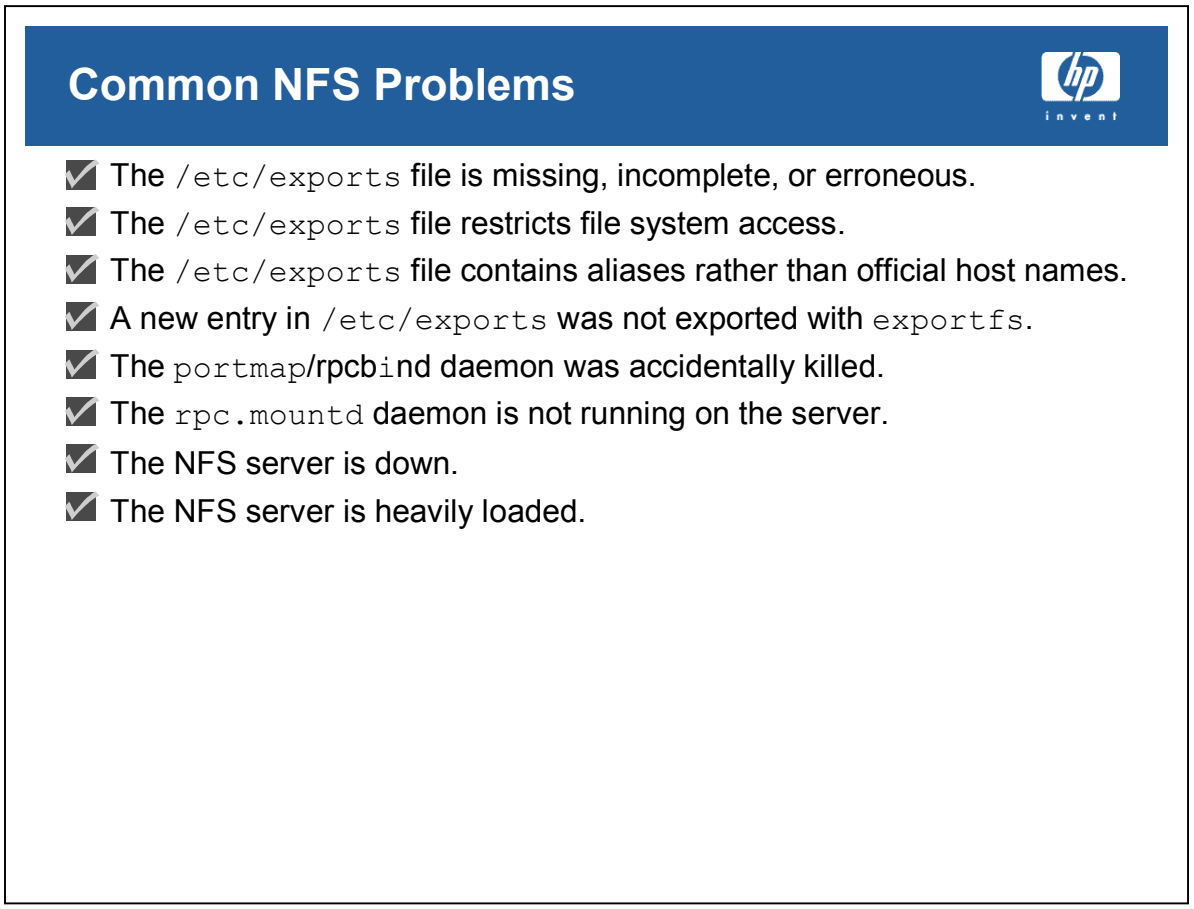
1. Keep UIDs and GIDs consistent.
2. Configure the NFS server.
   a. Ensure the NFS subsystem is in the kernel.
   b. Edit the server's configuration file.
   c. Start NFS server daemons.
   d. Create the `/etc/exports` file.
   e. Export the directories.
   f. Check the server configuration.
3. Configure the NFS client.
   a. Ensure the NFS subsystem is in the kernel.
   b. Edit the client's configuration file.
   c. Start NFS client daemons.
   d. Create a new entry in `/etc/fstab`.
   e. Mount the NFS file system.
   f. Check the client configuration.
4. Keep the time synchronized with all other nodes.

Student Notes

This slide is a review of all of the NFS configuration steps that we have already discussed.

© 2003 Hewlett-Packard Development Company, L.P.

9–17. SLIDE: Common NFS Problems



Student Notes

NFS has proven to be a stable, reliable mechanism for sharing files between UNIX hosts for over 15 years. However, most NFS administrators still inevitably need to do some NFS troubleshooting at some point. This slide highlights some of the most common NFS problems and misconfigurations.

- **`/etc/exports`** is missing, incomplete, or erroneous. Verify that the file system your client is trying to mount is included in the **`/etc/exports`** file with appropriate export options. Watch for invisible characters (control sequences) and invalid combinations of export options. If possible, use only the tested combinations of export options that were discussed in Tables 1 and 2 earlier in the chapter.

- **`/etc/exports`** restricts file system access. Try executing the **`showmount -e`** command on the NFS server to determine which clients are allowed to mount your server's file systems. If your client is not listed, you may need to modify the export options in **`/etc/exports`**.

- **`/etc/exports`** contains the alias of an NFS client instead of its official host name. NFS uses reverse name resolution to resolve clients' IP addresses into hostnames, then looks

ã 2003 Hewlett-Packard Development Company, L.P.

for the clients' hostnames in the export list. Be sure to use official hostnames in **/etc/exports**, not hostname aliases!

- The administrator added a new entry to **/etc/exports** without activating it with **exportfs**. Every time you modify **/etc/exports**, you must notify **rpc.mountd** that the export list changed by executing **exportfs -a**.

- The **portmap/rpcbind** daemon was accidentally killed. NFS uses RPC calls, and RPC calls are all handled initially by the **portmap/rpcbind** daemon. Without this daemon, NFS will not function properly! Check the process table to verify that the daemon is running. If the daemon is missing from the process table, you will have to stop and restart the NFS server and client daemons with **/sbin/init.d/nfs.server** and **/sbin/init.d/nfs.client**.

- The **rpc.mountd** daemon is not running on the server. Clients cannot mount file systems if **rpc.mountd** is not running on the server. Try running the **/sbin/init.d/nfs.server** program with the **start** argument to restart the daemon.

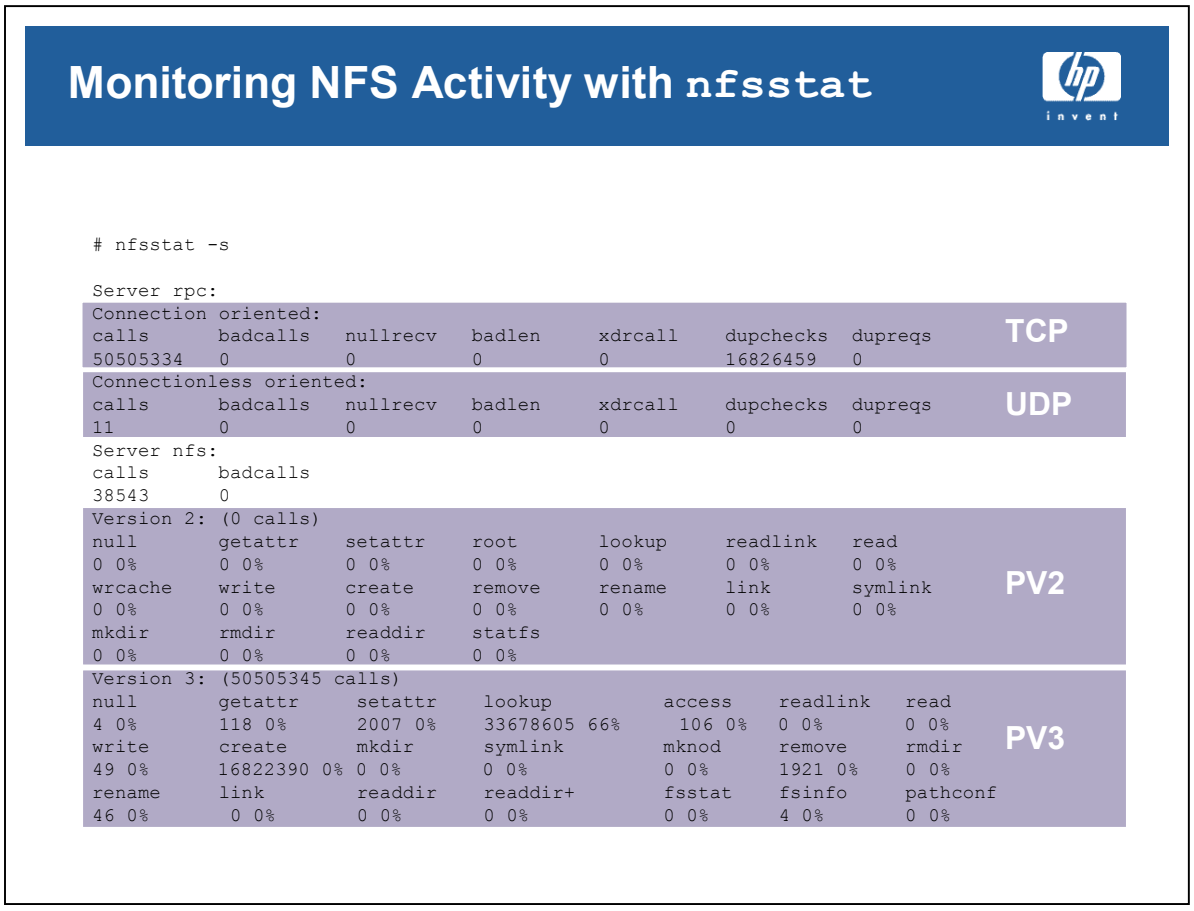
- The NFS server is down. Try to **ping** the remote system to check for network connectivity. If you can **ping** the system, but you cannot **mount**, the remote system may not have the proper daemons running. Try stopping and restarting NFS on the remote system. If you cannot ping the remote system, turn back to the Troubleshooting Network Connectivity chapter earlier in this book.

- The NFS server is heavily loaded. NFS performance will be degraded as the client/server ratio increases. Eventually, the server's performance may be degraded so much that client requests time out and fail. You can check this with the **nfsstat** command. There are several possible solutions to this problem:

  - Upgrade your NFS server.
  - Create an additional server and balance the load.
  - Increase the number of your NFS daemons (**nfsd**) on the server. It is recommended that the number of NFS daemons increase with the number of NFS clients.

© 2003 Hewlett-Packard Development Company, L.P.

9–18. SLIDE: Monitoring NFS Activity with `nfsstat`



Student Notes

Over time, you may wish to monitor the volume and type of NFS/RPC traffic on your network. This may help you troubleshoot performance problems and plan for future growth. You can use the **`nfsstat`** command to view the contents of several NFS registers maintained by the kernel. The **`-z`** option makes it possible to reinitialize these registers.

`-c` Displays client RPC requests only.

`-s` Displays server information.

`-n` Displays NFS information, but excludes general RPC statistics from the report.

`-m` Displays statistics for each NFS mounted file system. This includes the server name and address, mount flags, current read and write sizes, the retransmission count, and the timers used for dynamic retransmission.

`-r` Displays RPC information, but excludes NFS specific statistics.

ã 2003 Hewlett-Packard Development Company, L.P.

-z Prints the current statistics, then reinitializes them (resets them to zero). Combine **-z** with any of the options to reinitialize particular sets of statistics after printing them. The user must have write permission on **/dev/kmem** for this option to work.

The packet traffic via NFS is cumulatively monitored. Look especially for non-zero entries in the following fields. They indicate errors, called failures or timeouts:

badcalls
nullrecv
badlen
retrans
badxid
timeout

Many administrators configure a **cron** job to automatically execute **nfsstat -z** on a weekly or monthly basis. **nfsstat -z** displays all of the current values and then zeroes out the registers. Comparing these reports makes it possible to track your NFS usage over time. If you notice uncommonly high values for "Server rpc" and "Server nfs," your system may be overloaded as the server.

© 2003 Hewlett-Packard Development Company, L.P.

9–19. LAB: Configuring NFS

Directions

In this lab, you will work with a partner to experiment with some of the features of NFS. One of you will function as an NFS server, and the other will function as an NFS client. You should work *together* throughout the lab to ensure that you feel comfortable with both the client and server functionalities of NFS. At this point, decide between yourselves who will be the server and who will be the client.

Host name of server: _____

Host name of client: _____

Preliminary Steps

1. **(client)**

   Install the lab files needed on your client:

   ```
   # cd /labs
   # tar -xvf nfs.client.tar
   ```

   You should now have two new user accounts defined in your **`/etc/passwd`** file: "mickie" and "minnie." The passwords for the new accounts are "mickie" and "minnie" respectively. Note that neither user has a home directory on your machine. You will mount their home directories from your partner's NFS server.

2. **(server)**

   Install the lab files needed on your server:

   ```
   # cd /labs
   # tar -xvf nfs.server.tar
   ```

   This tarball creates several new files and directories, and two new user accounts in your **`/etc/passwd`** file for users "mickie" and "minnie." The passwords for the new accounts are "mickie" and "minnie" respectively. The tarball also creates home directories for mickie and minnie.

ã 2003 Hewlett-Packard Development Company, L.P.

Part 1: Basic NFS Configuration

1. **(client and server)**

   In order for NFS to function properly, the InternetSrvcs and Networking products must be installed on your machine. Check to ensure that both of these products have been installed on your machine. Also, ensure that the NFS subsystem is configured in the kernel.

   ```
   # swlist -l product Networking InternetSrvcs NFS
   # grep nfs /stand/system
   ```

2. **(client and server)**

   Is your machine configured as an NFS server, client, or both?

   What configuration file should you check to find out?

   Make sure the appropriate functionality is configured.

3. **(client)**

   What daemons should you see on an NFS client?

   Use **`ps -e`** on the client to ensure that the necessary daemons are actually running.

4. **(server)**

   What daemons should you see on an NFS server?

   Use **`ps -e`** to ensure that the server has the necessary daemons running.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Exporting and Mounting NFS File Systems

1. **(server)**

   Your clients need to access several files on your server machine. Export the following with the export options set as noted. Make the file systems available to clients immediately, but also ensure that they will be available after the next system boot by adding them to **`/etc/exports`**.

```
/home          rw for your partner's machine, no access for other hosts
/opt/phone     rw for your partner's machine, read only for all others
/opt/fun       read only for everyone on the LAN
```

2. **(server)**

   What command can you use to see what file systems you have made available? Can you tell which export options you used?

   What command can you use to see what file systems other servers have made available? Choose another machine in the classroom and see what it has exported. Can you tell which export options were used?

3. **(client)**

   Create mount points for the file systems your neighbor exported, and mount them:

```
/home/mickie
/home/minnie
/opt/fun
/opt/phone
```

4. **(client)**

   What file needs to be modified to ensure that these NFS file systems are automatically mounted after every system boot? Make it so. (For now, use the "defaults" mount option.)

   Syntax errors in the **`/etc/fstab`** file may cause the next system boot to fail. Do a **`mount -a`** to ensure that you did not make any mistakes in **`fstab`** file.

ã 2003 Hewlett-Packard Development Company, L.P.

Finally, use **`mount -v`** to ensure that all the NFS file systems actually mounted properly.

5. **(server)**

   What command lists the remote machines that have your exported file systems mounted?

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Using NFS File Systems

1. **(client and server)**

   Some shops use NFS to export file systems containing application executables. This offers a number of benefits. You only need to allocate disk space for the application on the NFS server, not on every client. It also simplifies upgrades, since the application is stored in just one place.

   From your client, try executing some of the programs mounted from the NFS server to verify that this is true:

   ```
   client# /opt/fun/melt
   client# /opt/fun/xroach -speed 1
   ```

   Another benefit of NFS is that files created in an NFS file system instantly become available to multiple client machines. Do the following experiment to verify that this is true:

   ```
   client# ls /home/mickie
   server# touch /home/mickie/data
   client# ls /home/mickie
   ```

   Does the client see the new file that was created on the server?

2. **(client and server)**

   Though access to files shared via NFS should be more or less transparent to your users, file access restrictions can mean that a user is able to access a file on some machines but not others. Try the following commands while logged on as root:

   ```
   client# cp /opt/fun/melt /opt/fun/drip
   server# cp /opt/fun/melt /opt/fun/drip
   ```

   Why did this command succeed when executed on the server, but not when executed on the client? (hint: look at **`/etc/exports`**)

3. **(client)**

   Let's try a variation on the experiment you did back in Q#1 of this part of the lab.

   ```
   client# touch /home/mickie/memo
   ```

   Why did this fail?

ã 2003 Hewlett-Packard Development Company, L.P.

Was the file system exported with "ro" permission?

Was the file system mounted with "ro" permission?

As root, shouldn't you be able to create **`/home/mickie/memo`**?

Do whatever is necessary to successfully execute the touch command on the client. (You should not have to type anything on the server. Hint: Which user on the client has write permission on mickie's home directory?)

4. **(client and server)**

We saw in the previous question that root on an NFS client does not (by default) have the same file access as root on the NFS server. If a single administrator manages several systems, however, it may be useful to allow root on NFS clients to have true root access to exported file systems.

What would you have to do on the NFS server side to allow root on the client to have the same full root access to the **`/home`** file system? Make it so.

Did this seem to work? While logged in as root on the client, try touching a file in mickie's home directory. Did you have to do anything on the client side to recognize the change in the server's **`exports`** file?

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Unmounting NFS File Systems

1. **(client)**

   Occasionally, it becomes necessary to unmount file systems to perform some administrative tasks. Let's start with the easiest case: on the client machine, unmount **/home/mickie**.

   Use **mount -v** to see which file systems remain in the client's mount table. Also do an **ls** of **/home/mickie**, and note that the memo and data files that were under **/home/mickie** no longer appear since the file system has been unmounted.

2. **(client)**

   Let's try a more complicated scenario. Can the client unmount an NFS file system if one of the client's users is accessing that file system?

   On the client machine, open two windows. In one of the windows, **cd** to the **/home/minnie** directory. In the other window, issue the **umount** command to unmount the minnie file system. Did this work?

   The **fuser** command can tell you who is currently using a file system. Try the following to see who is currently using **/home/minnie**.

   ```
   client# fuser -cu /home/minnie
   ```

   Try a **fuser -cuk** on **/home/minnie**, and see what happens. Then try unmounting the file system again.

3. **(server)**

   In Part 2, Question 5, we saw a command that the server administrator could use to determine which of the exported file systems were actually mounted on client hosts. Now try executing that command again. Was the NFS server notified when the client unmounted mickie and minnie?

ã 2003 Hewlett-Packard Development Company, L.P.

4. **(server and client)**

   We saw that the administrator can force users out of a mounted file system with the **`fuser`** command. If **`fuser`** is executed on the NFS server, does it kill processes on the NFS clients, or just on the server itself? Try it.

   ```
   client# cd /opt/fun
   server# fuser -cuk /opt
   ```

   Unfortunately, there is no mechanism in NFS to kill client processes from the server.

5. **(server and client)**

   We just discovered in the previous question that the NFS server has no way of killing processes on client hosts. Local file systems cannot be unmounted until all processes using them die. Does this mean that an NFS server administrator is unable to unmount his/her exported file systems until the clients that have mounted those file systems voluntarily unmount? Let's find out.

   ```
   server# fuser -cuk /opt  # kill any proc's on the svr using /opt
   server# umount /opt      # unmount the local /opt file system
   ```

   Did you successfully unmount the file system? Any errors? What happened to the client process that was using your exported **`/opt`**?

   Try the following commands on the client and note the output.

   ```
   client# pwd
   client# ls
   client# cd ..
   client# cd /
   client# umount /opt/fun
   ```

   On the client, could you unmount **`/opt/fun`**, even after the server was unmounted?

6. **(server and client)**

   Summarizing what you saw in the previous question: If an NFS server unmounts an exported file system that a client has mounted:

   a. Can the client still access files in the affected file system?
   b. What happens to client processes accessing the affected file system?
   c. Can the client unmount the imported file system?

© 2003 Hewlett-Packard Development Company, L.P.

7. **(server and client)**

   Remount all the server and client file systems on both the server and client.

ã 2003 Hewlett-Packard Development Company, L.P.

Part 5: (Optional) When Things Go Wrong

1. During the remainder of the lab, you will be asked to shutdown your LAN card several times. Execute the following command to shutdown CDE before proceeding:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. **(server and client)**
   What happens if the NFS client loses LAN connectivity to the server? Do the following and note the output from the commands. First, note the client's behavior when the server is up. (It should be normal.)

   ```
   client# cd /opt/fun
   client# ls
   ```

   Now take the server's LAN card down and note what happens to the client:

   ```
   server# ifconfig lan0 down
   client# ls                    #This will hang indefinitely
   ```

   Move on to next step.

   What happens when the client regains connectivity to the NFS server?

   ```
   server# ifconfig lan0 up
   client# ls
   ```

3. **(server and client)**
   So, what can the client administrator do while the NFS server is down? Can the client administrator unmount the NFS file system? Try it.

   ```
   server# ifconfig lan0 down
   client# cd /
   client# umount /opt/fun       # Be patient.
   client# mount
   ```

   What happens if the client tries to remount that file system again while the server is still down? Try it.

   ```
   client# mount /opt/fun        # Be patient.
   ```


© 2003 Hewlett-Packard Development Company, L.P.

4. **(server and client)**
   Hopefully you discovered that a client can always unmount an NFS file system, even if the NFS server is down. Actually, since NFS is a "stateless" system, the server can always unmount its local file systems, too, even if clients have them mounted. Of course doing so will cause problems for the clients.

   To summarize, when an NFS server goes down...

   Are any of the processes on the client killed?

   What happens when a process on the client tries to hit a file system on the downed server (assuming the default mount options are used)? Do they hang indefinitely or time out?

   What happens when a client tries to mount a file system from a downed server? (Again, assume that the default mount options are used.)

5. **(server and client)**

   Bring the server and client back to their original states:

```
server# ifconfig lan0 up
server# mount -a
server# exportfs -a
client# mount -a
```


ã 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Client Side Mounting Options

1. **(server and client)** **`intr`** and **`nointr`** mount options

   By default, HP-UX mounts NFS file systems "hard,intr." If the NFS server goes down with these default mount options, we saw client attempts to access the NFS files and directories hang indefinitely. Can the user abort a command if they get tired of waiting? Try it.

   ```
   server# ifconfig lan0 down
   client# ls /opt/fun      # can the user abort the ls with ^C?
   server# ifconfig lan0 up
   ```

   Alternatively, you can mount an NFS file system **`nointr`**. How would the **`nointr`** mount option affect the experiment above? Try it.

   ```
   client# umount /opt/fun
   client# mount -o nointr server:/opt/fun /opt/fun
   server# ifconfig lan0 down
   client# ls /opt/fun      # can the user abort the ls with ^C?
   ```

   When will the user get a prompt back?

2. **(server and client)** Soft versus Hard Mounts

   The client can also override the **`hard`** option with **`mount -o soft`**. If a client has mounted an NFS file system "soft" and the NFS server goes down, what happens to client requests to the server? Try it.

   ```
   server# ifconfig lan0 up
   client# umount /opt/fun
   client# mount -o soft server:/opt/fun /opt/fun
   server# ifconfig lan0 down
   client# ls /opt/fun        # be patient.
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 7: (Optional) Troubleshooting a Non-responsive NFS Server

1. **(client and server)**

   You have seen the effect that a downed NFS server has on NFS clients. What can the administrator on the client side do to determine what might be wrong on the server side? Do an experiment to find out. Start by doing some experiments while both your server and client are functioning properly. Bring the LAN cards on both machines to an "UP" state:

   ```
   server# ifconfig lan0 up
   client# ifconfig lan0 up
   ```

   Now test connectivity from the client to your NFS server:

   ```
   client# ping server
   client# rpcinfo -p server
   ```

2. **(server and client)**
   Now shut down NFS on your NFS server.

   ```
   server# /sbin/init.d/nfs.server stop
   ```

   From the client, try mounting **/opt/fun**.

   ```
   client# mount /opt/fun # After you see the error, hit ^C.
   ```

3. **(client)**
   From the client, try your connectivity test commands again:

   ```
   client# ping server
   client# rpcinfo -p server
   ```

   Can you still **ping** the server?

   Which RPC programs are no longer available on the server?

   Will clients be able to mount NFS file systems now? Why?

   Will clients be able to access already mounted NFS file systems? Why?

ã 2003 Hewlett-Packard Development Company, L.P.

Part 8: Cleanup

1. Before moving on to the next chapter, restore your network configuration to the state it was in prior to this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

ã 2003 Hewlett-Packard Development Company, L.P.

Module 10 — Configuring AutoFS

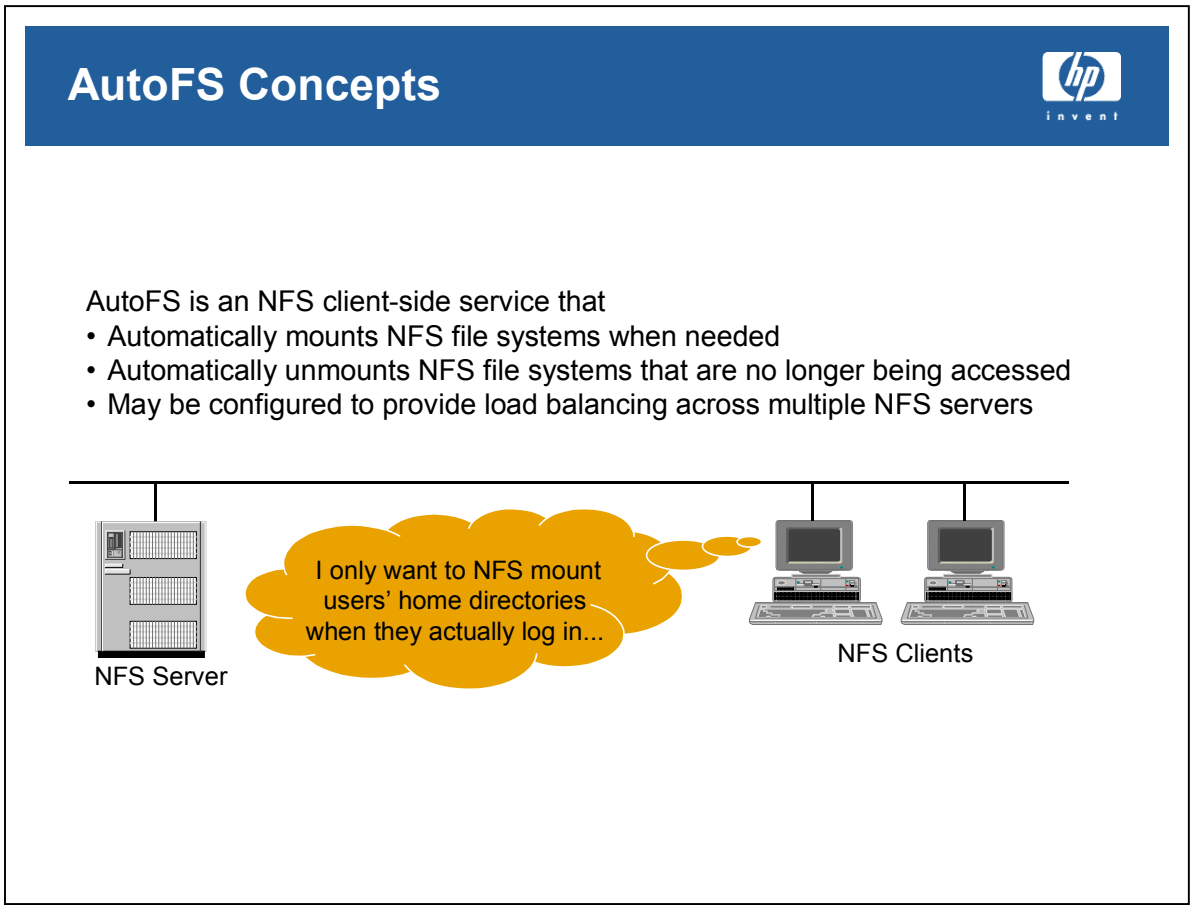
Objectives

Upon completion of this module, you will be able to do the following:

- Describe the reasons for using AutoFS.
- Start and stop the AutoFS daemons.
- Configure the AutoFS master map.
- Configure the AutoFS **`-hosts`** special map.
- Configure the AutoFS direct map.
- Configure the AutoFS indirect maps.
- Describe the differences between AutoFS direct and indirect maps.
- Configure AutoFS to mount and unmount user home directories.
- Troubleshoot problems with AutoFS.
- Identify the limitations of AutoFS's predecessor, the NFS Automounter.

© 2003 Hewlett-Packard Development Company, L.P.

10–1. SLIDE: AutoFS Concepts



Student Notes

The Limitations of NFS

You learned in the previous chapters that NFS provides a convenient mechanism for sharing files and directories across a local area network. Many administrators use NFS to share executables, data files, and even home directories among multiple hosts on their LANs. However, administrators that use NFS extensively are likely to encounter a number of limitations:

- In order to ensure that an NFS file system is available after every system boot, the file system must be added to the **`/etc/fstab`** file. As more and more NFS file systems are added to **`/etc/fstab`**, the file becomes unwieldy.

- Maintaining complex NFS mounts in the **`/etc/fstab`** files on multiple clients can quickly become a support nightmare.

- If an NFS server referenced in **`/etc/fstab`** is unavailable during an NFS client's boot process, the client hangs temporarily until the mount request times out. As more and

© 2003 Hewlett-Packard Development Company, L.P.

more NFS file systems are added to **`/etc/fstab`**, the chance of an NFS time-out occurring during the boot process increases dramatically.

- Only root can mount NFS file systems. If a user needs to temporarily mount an NFS file system on a client, the user must ask the administrator to mount and unmount the file system for them.

The Advantages of AutoFS

AutoFS is an NFS client-side service designed to address all of the limitations mentioned above – and more!

- AutoFS automatically mounts NFS file systems on an as-needed basis. File systems managed by AutoFS can be removed from **`/etc/fstab`**, making the file much less cumbersome.

- The AutoFS configuration files, known as the AutoFS "maps," can be managed via NIS. Instead of managing **`/etc/fstab`** files on hundreds of individual hosts, the administrator can easily modify the NFS configuration from the central NIS server that stores the NIS AutoFS maps.

- AutoFS only mounts NFS file systems on an as-needed basis. Thus, a downed NFS server will only delay a client's boot if the client references the downed server's file systems during the boot process.versusversus

- AutoFS may be configured to allow users to automatically mount available NFS file systems without root's assistance.

- By default, if an AutoFS file system is left unused for five minutes, AutoFS automatically unmounts the file system.

- AutoFS provides some primitive load balancing across multiple replicated NFS servers. If an NFS file system is available from several different servers, AutoFS will automatically mount the file system from the server that provides the best response time.

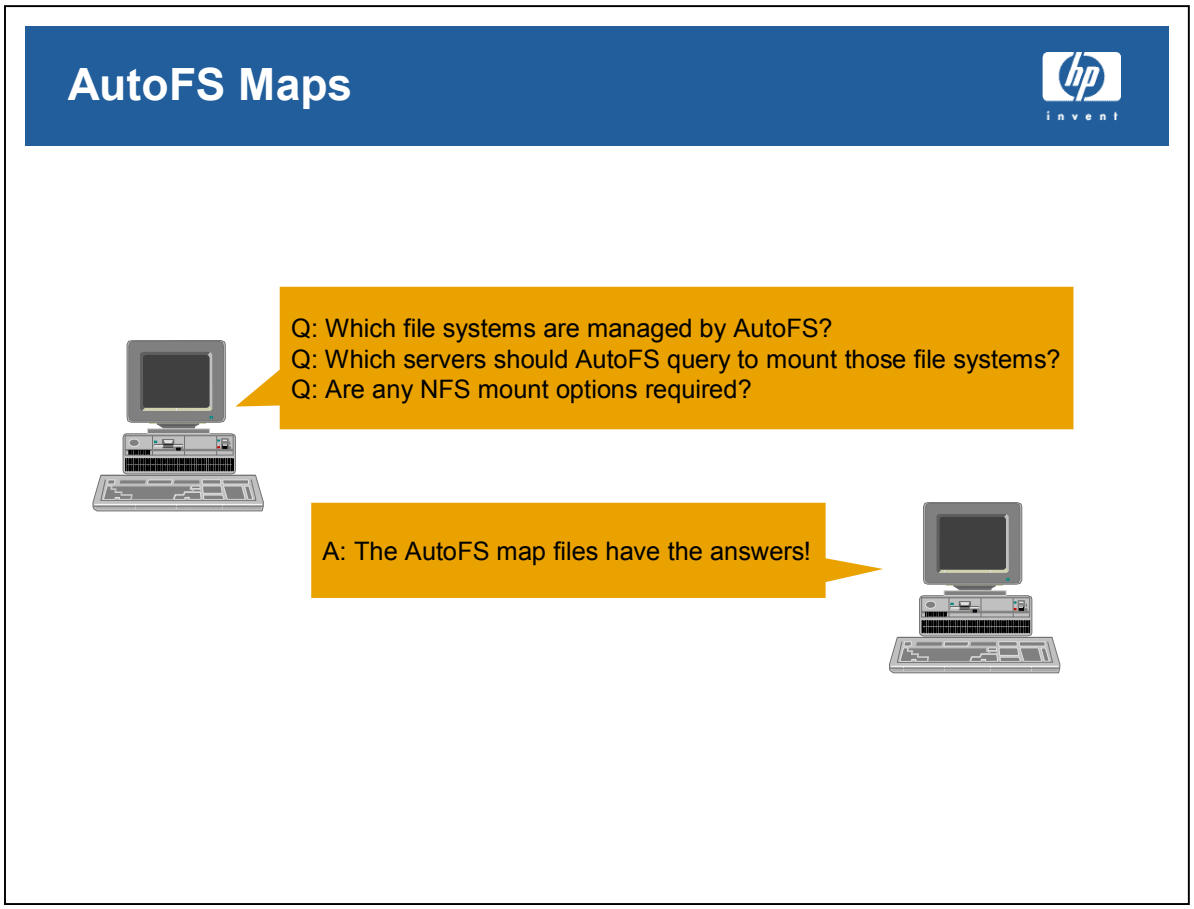
AutoFS versus Automounter

Before HP-UX version 10.20, HP's NFS implementation included Automounter rather than AutoFS. Although both services provide similar functionality, AutoFS is more robust. Versions 10.20 and 11.x include both services, but HP has stated that future releases of the operating system will only support AutoFS. For more information on the differences between the two services, see the slide at the end of this chapter.

NOTE:	AutoFS simply generates NFS mount and unmount requests on behalf of an NFS client. AutoFS can only mount file systems that have been exported by an NFS server.

© 2003 Hewlett-Packard Development Company, L.P.

10–2. SLIDE: AutoFS Maps



Student Notes

NFS file systems may be mounted via the **`mount`** command, or via AutoFS. When **`/sbin/init.d/nfs.client`** executes the **`mount`** command during the boot process, it immediately mounts all of the NFS file systems listed in **`/etc/fstab`**.

AutoFS, however, mounts NFS file systems on an as-needed basis. In order to do this, AutoFS must be told:

- Which file systems to mount;
- Which NFS servers provide those file systems; and
- Which mount options should be used when mounting those file systems.

The AutoFS **map files** answer all three questions. The map files are ASCII configuration files managed by the system administrator. You may use the **`ls`** command to view the AutoFS maps (if there are any!) on your system:

```
# ls /etc/auto*
```

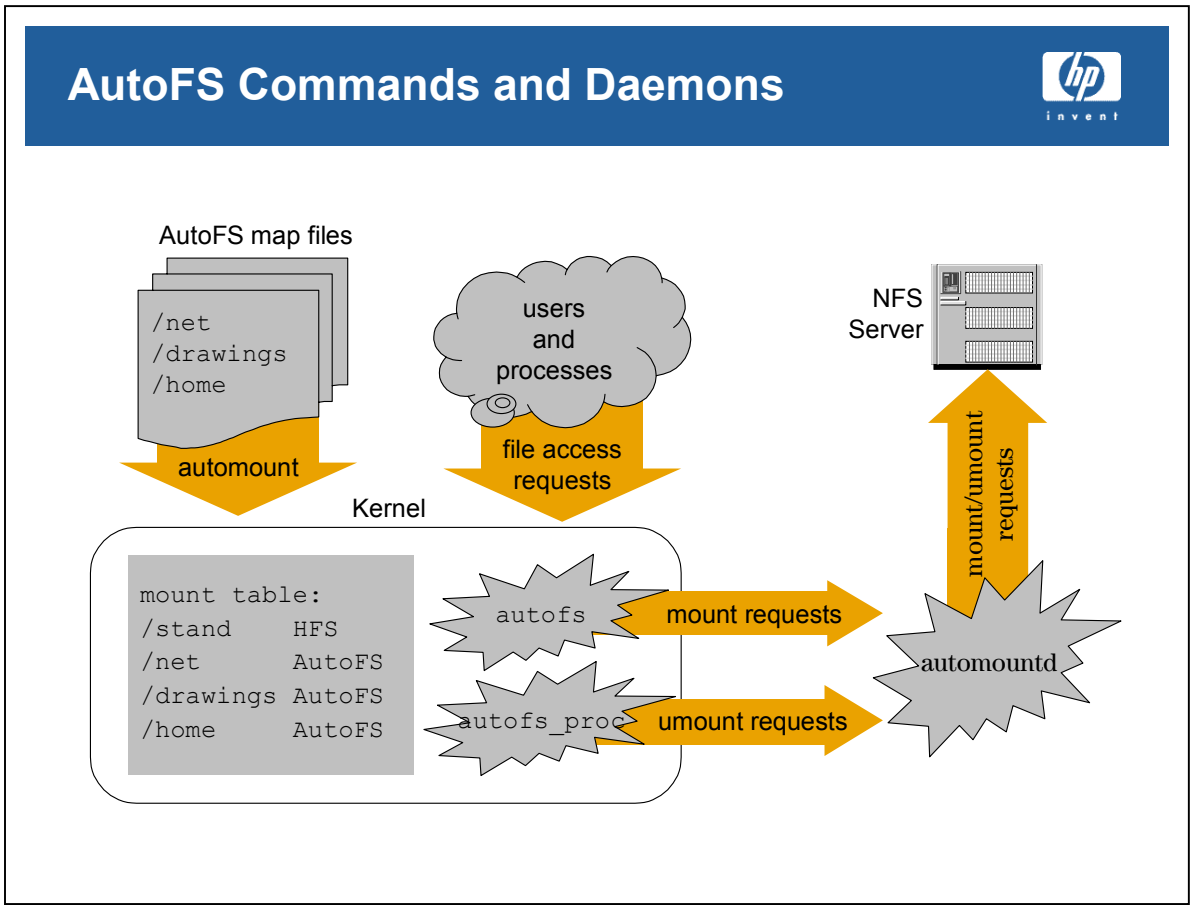
Some AutoFS map files on your systems may be managed via NIS. These NIS-managed map files won't appear in the **`ls`** output.

© 2003 Hewlett-Packard Development Company, L.P.

AutoFS recognizes several different kinds of map files. Each of these maps will be discussed in detail in the slides that follow.

© 2003 Hewlett-Packard Development Company, L.P.

10–3. SLIDE: AutoFS Commands and Daemons



Student Notes

AutoFS requires several different daemons and commands:

1. The first step required to configure AutoFS is to create the AutoFS map files. The next few slides discuss the configuration of these files in detail.

2. Anytime you modify the AutoFS map files, you must execute the **`automount`** command. This command reads the AutoFS maps, then adds and removes AutoFS entries in the **`/etc/mnttab`** mount table accordingly. Note that **`automount`** doesn't actually mount any file systems; it is simply responsible for ensuring that the AutoFS entries in the mount table match the AutoFS maps.

3. When processes attempt to access the AutoFS file systems recorded in the mount table, AutoFS contacts the **`automountd`** daemon.

4. When AutoFS notifies the **`automountd`** daemon that an NFS file system is required, **`automountd`** sends an NFS mount request to the appropriate NFS server.

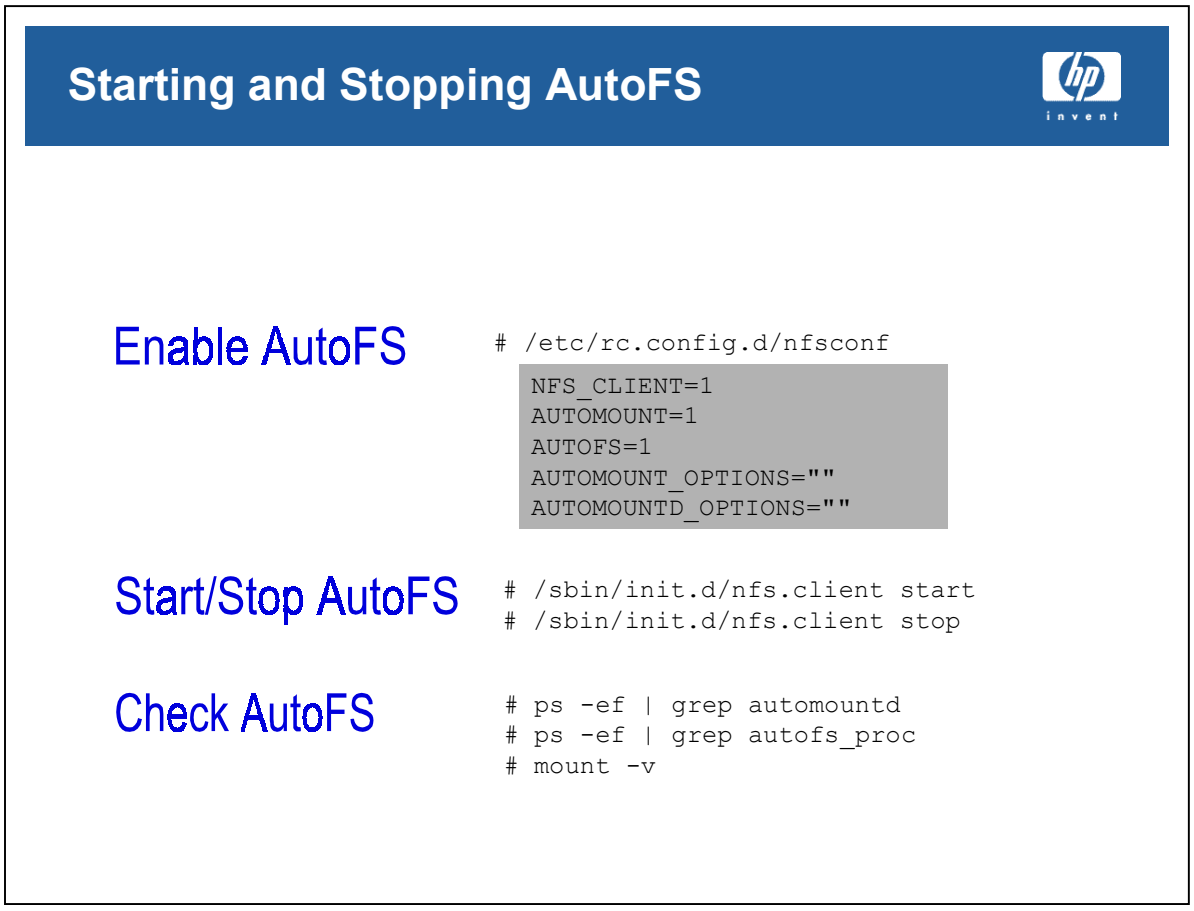
5. Once **`automountd`** mounts the needed file system, the requesting process can access the file system as it would any other NFS file system.

© 2003 Hewlett-Packard Development Company, L.P.

6. The **`autofs_proc`** kernel daemon monitors all NFS file systems mounted by AutoFS. If an NFS file system managed by AutoFS is idle for 5 minutes, **`autofs_proc`** notifies **`automountd`**, which then unmounts the idle file system. The allowed idle time is configurable. This prevents unnecessary NFS file systems from cluttering the mount table.

© 2003 Hewlett-Packard Development Company, L.P.

10–4. SLIDE: Starting and Stopping AutoFS



Student Notes

AutoFS is an NFS client-side service. No additional server-side configuration is required, beyond enabling the **`nfsd`** and **`rpc.mountd`** daemons, and exporting the desired file systems.

Enabling AutoFS Functionality

In order to run AutoFS on an NFS client, several variables must be set in **`/etc/rc.config.d/nfsconf`**. First, verify that basic NFS client functionality is enabled:

```
NFS_CLIENT=1
```

Next, verify that the **`AUTOMOUNT`** variable is set to "1". Although the **`AUTOMOUNT`** variable was traditionally used to enable the old **`automount`** daemon, it is still required if you wish the newer AutoFS daemons to start during the system boot process.

© 2003 Hewlett-Packard Development Company, L.P.

To specify that you wish to use AutoFS rather than the traditional Automounter, scroll to the bottom of the file and set the **AUTOFS** variable, too.

```
AUTOMOUNT=1
AUTOFS=1
```

The last couple of variables may be used to define additional options for the AutoFS daemons:

```
AUTOMOUNT_OPTIONS=""
AUTOMOUNTD_OPTIONS=""
```

A table describing some of the commonly used options available for these variables is included below. For more information, see the **automount(1m)** and **automountd(1m)** man pages.

AUTOMOUNT_OPTIONS="-t 600"	By default, AutoFS automatically unmounts file systems that have been idle for 300 seconds (5 minutes). You may increase the allowed idle time via the **AUTOMOUNT_OPTIONS** variable.
AUTOMOUNT_OPTIONS="-v"	Verbose. Displays a message to stdout when the AutoFS configuration changes.
AUTOMOUNTD_OPTIONS="-v -T"	Enable verbose logging of all AutoFS mount and umount requests in **/var/adm/automount.log**.

Starting AutoFS

If the **AUTOFS** variable is set to "1" in **/etc/rc.config.d/nfsconf**, then AutoFS is normally started automatically by the **/sbin/init.d/nfs.client** script at run level 2 of the system startup process. You may re-execute this script at any time:

```
# /sbin/init.d/nfs.client start
```

Running the script with the **start** argument mounts all NFS file systems in **/etc/fstab** and starts the AutoFS daemons.

Although AutoFS is usually started by the **/sbin/init.d/nfs.client** script, you can manually start the service by issuing two commands:

```
# /usr/lib/netsvc/fs/autofs/automountd
# /usr/sbin/automount
```

The first command starts the **automountd** daemon that generates mount requests to the NFS server. The second command copies the AutoFS map information into **/etc/mnttab** so **automountd** knows which file systems it is responsible for mounting.

NOTE: AutoFS and Automounter cannot run concurrently on an NFS client. If you are currently using Automounter, modify the **/etc/rc.config.d/nfsconf** configuration file as shown on the slide, then reboot to stop the currently running Automounter daemon and start AutoFS.

© 2003 Hewlett-Packard Development Company, L.P.

Stopping AutoFS

Usually, AutoFS is terminated by **/sbin/init.d/nfs.client** during system shutdown:

```
# /sbin/init.d/nfs.client stop
```

Alternately, you can manually shutdown AutoFS with the following commands:

```
# ps -e | grep automountd
# kill 1234                    Use the automountd daemon's PID here!
# /usr/sbin/umountall -F nfs
# /usr/sbin/umountall -F autofs
```

If a file system mounted by AutoFS is still in use when the stop script is executed, that file system remains mounted and must be manually unmounted later by issuing the **umountall** commands shown above.

NOTE: *Never* kill the **automountd** daemon with the **-9** signal! This will leave AutoFS in an inconsistent state, and may eventually require a reboot.

Checking AutoFS

If AutoFS is functioning properly, two daemons should appear in your process table: **automountd** and **autofs_proc**:

```
# ps -e | grep automountd
# ps -e | grep autofs_proc
```

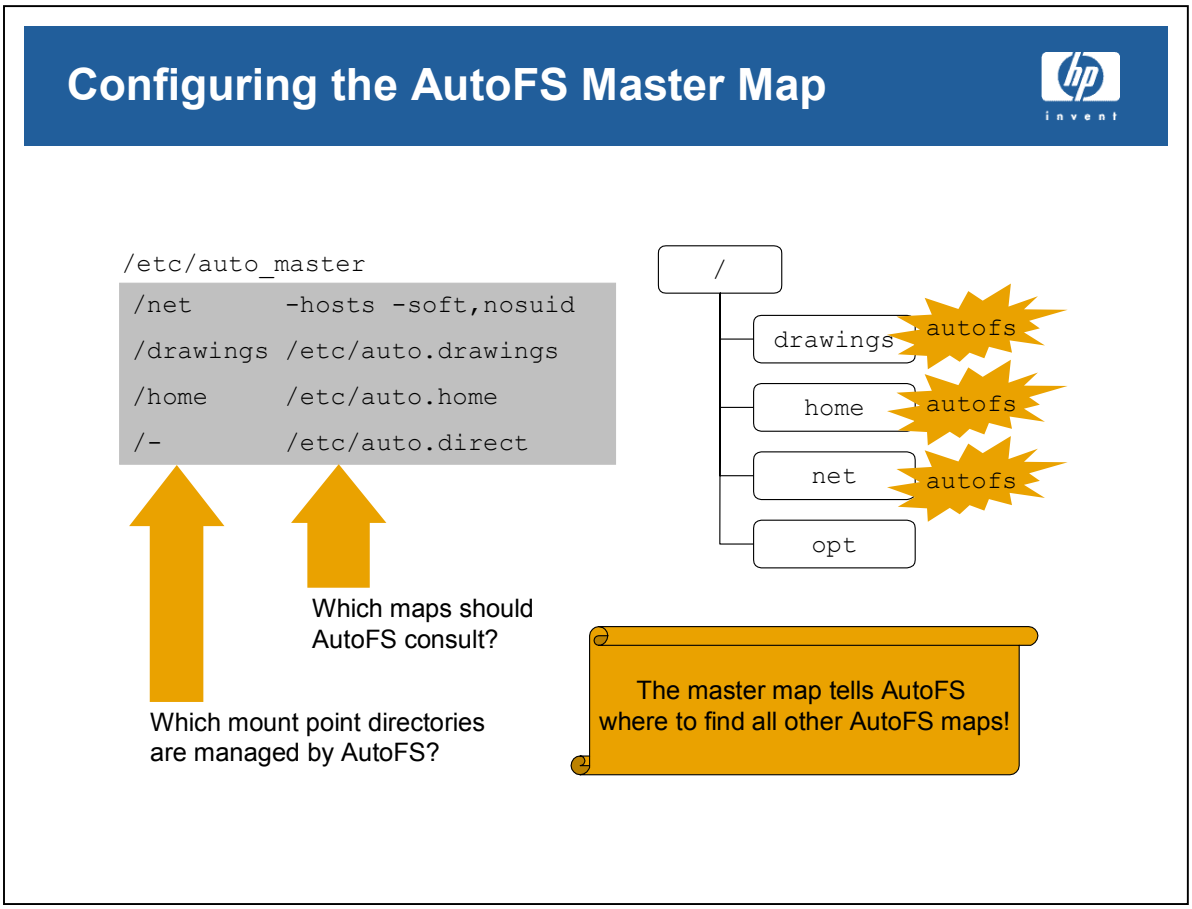
Also, check the mount table via the **mount -v** command. There should be an entry for each of the file systems managed by AutoFS. If not, check your map files! The sample **mount -v** output below was taken from a host that uses AutoFS extensively. Note: Local file systems and mount timestamps have been truncated to save space.

```
# mount -v
-hosts on /net type autofs ignore,indirect,nosuid,soft
/etc/auto.direct on /usr/contrib/games type autofs ignore,direct
/etc/auto.direct on /opt/tools type autofs ignore,direct
/etc/auto.direct on /var/mail type autofs ignore,direct
/etc/auto.drawings on /drawings type autofs ignore,indirect
/etc/auto.home on /home type autofs ignore,indirect
```

If AutoFS appears to be misbehaving, enable AutoFS logging in **/etc/rc.config.d/nfsconf**, stop and restart the service, and check the **/var/adm/automount.log** log file for errors.

© 2003 Hewlett-Packard Development Company, L.P.

10–5. SLIDE: Configuring the AutoFS Master Map



Student Notes

The AutoFS maps determine which file systems AutoFS should mount from which NFS servers. **`/etc/auto_master`** is a special map: it contains a catalog of mount point directories, followed by the names of the maps AutoFS should consult to determine what should be mounted under those directories.

The sample **`/etc/auto_master`** file on the slide references several other AutoFS maps:

- Attempts to access anything under **`/net`** will be handled by the special **`-hosts`** map.
- Attempts to access anything under **`/drawings`** will be handled by the **`/etc/auto.drawings`** map.
- Attempts to access anything under **`/home`** will be handled by the **`/etc/auto.home`** map.
- The **`/-`** entry at the end of **`/etc/auto_master`** refers AutoFS to the “direct map” in **`/etc/auto.direct`**.

Each of these referenced maps will be discussed in detail in the slides that follow.

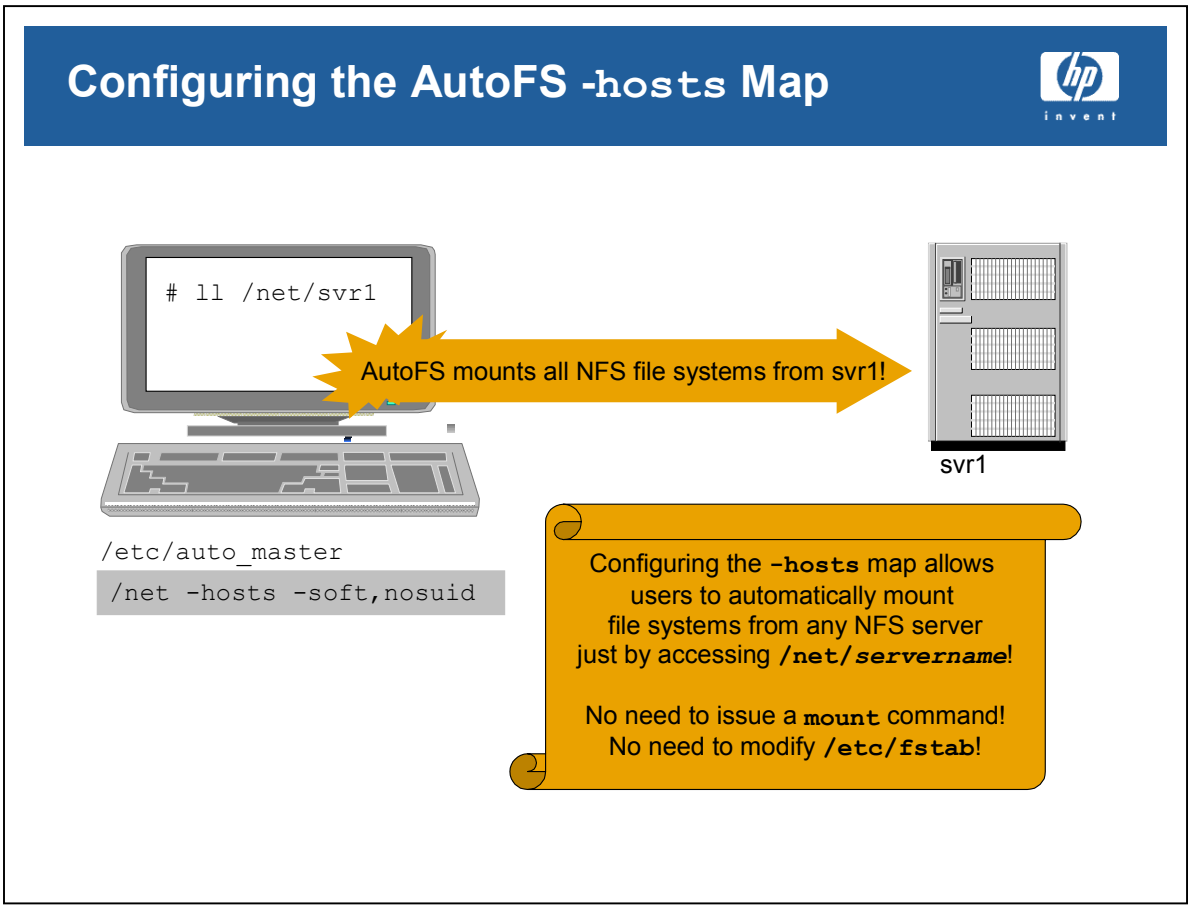
© 2003 Hewlett-Packard Development Company, L.P.

If **/etc/auto_master** doesn't exist when AutoFS is started, a minimal **/etc/auto_master** file is created automatically with just one map entry: "**/net -hosts -nosuid,soft**".

NOTE: Be sure to execute the **/usr/sbin/automount** command anytime you make changes to the master map!

© 2003 Hewlett-Packard Development Company, L.P.

10–6. SLIDE: Configuring the AutoFS –hosts Map



Student Notes

One of the most useful maps recognized by AutoFS is the **–hosts** special map. If **/etc/auto_master** is configured as shown on the slide, then accessing ***/net/any_NFS_server*** causes AutoFS to automatically mount all NFS file systems available to the client from the specified server. This makes it possible to mount all available NFS file systems from any NFS server without explicitly executing the **mount** command or modifying **/etc/fstab**!

Example

If the **–hosts** special map is configured as shown on the slide, you would see the following entry in your client's mount table initially (note that local file systems and the mount time stamps have been omitted for the sake of clarity).

```
# mount -v
  -hosts on /net type autofs ignore,indirect,soft,nosuid
```


© 2003 Hewlett-Packard Development Company, L.P.

At this point, if a user does an **ll** of the **/net** directory, nothing appears:

```
# ll /net
  total 0
```

See what happens, though, if a user accesses a specific host name within **/net**:

```
# ll /net/svr1
  dr-xr-xr-x   3 root sys  1024 Mar 28 08:50  home
  dr-xr-xr-x  44 bin  bin  1024 Mar 29 13:54  opt
  dr-xr-xr-x  18 bin  bin  1024 Mar 24 12:17  var
```

The output suggests that host svr1 has exported three NFS file systems: **/home**, **/opt**, and **/var**. Look what appears in the mount table as a result (again, the **mount -v** output has been truncated for the sake of clarity):

```
# mount -v
 -hosts on /net type autofs ignore,indirect,soft,nosuid
 svr1:/home on /net/svr1/home type nfs nosuid,soft,size=32768,NFSv3
 svr1:/opt on /net/svr1/opt type nfs nosuid,soft,rsize=32768,NFSv3
 svr1:/var on /net/svr1/var type nfs nosuid,soft,rsize=32768,NFSv3
```

Configuring the -hosts Special Map

In order to make the **-hosts** functionality available on your NFS client, verify that the following line is included in **/etc/auto_master**, then execute the **/usr/sbin/automount** command to force AutoFS to reread the maps.

```
# vi /etc/auto_master
  /net -hosts -soft,nosuid
```

The **-soft** NFS mount option prevents users' access attempts from hanging if the client is the NFS server is unreachable. The **nosuid** mount option is a security feature that disables the SUID bit execution for programs accessed from the NFS server.

NOTE: Be sure to execute the **/usr/sbin/automount** command after you add or remove the **-hosts** entry in **/etc/auto_master**.

Disadvantages of the -hosts Special Map

The **-hosts** map has just three disadvantages that you should be aware of.

- When a user accesses **/net/*any_NFS_server***, AutoFS mounts *all* of the NFS file systems available from the specified server. If frequent access to a single file system is required, it is more efficient to access the file system with a map entry that is tailored to mount just the file system of interest. The direct and indirect maps discussed on the next couple slides do just that.

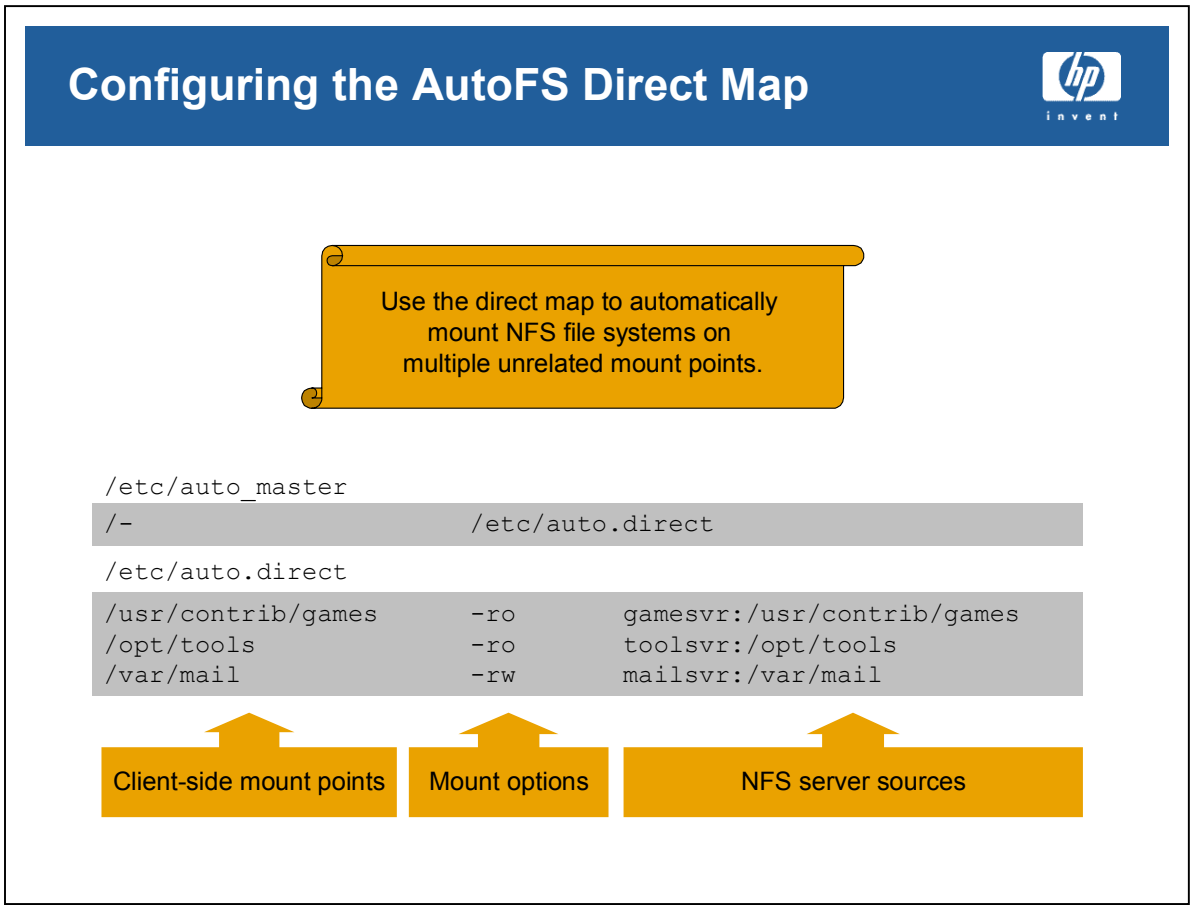
- If a user attempts to use **/net** to access an unreachable NFS server, or an NFS server that hasn't exported any NFS file systems, AutoFS generates a "not found" error condition, which may confuse your users.

© 2003 Hewlett-Packard Development Company, L.P.

- Because the **`-hosts`** map allows NFS access to any reachable system, a user may inadvertently cause an NFS mount over a WAN link, or through a slow router or gateway. NFS mounts over slow links may cause excessive retransmissions and degrade performance for all users on the network.

© 2003 Hewlett-Packard Development Company, L.P.

10–7. SLIDE: Configuring the AutoFS Direct Map



Student Notes

A direct map may be used to automatically mount file systems on any number of unrelated mount points.

The sample **`/etc/auto.direct`** file shown on the slide:

- Mounts **`/usr/contrib/games`**, read-only, from the gamesvr NFS server.
- Mounts **`/opt/tools`**, read-only, from the toolsvr NFS server.
- Mounts **`/var/mail`**, read-write, from the mailsvr NFS server.

Example

If the **`/etc/auto_master`** and **`/etc/auto.direct`** are configured as shown on the slide, you would see the following entry in your client's mount table initially (note that local file systems and the mount time stamps have been omitted for the sake of clarity).

```
# mount -v
  /etc/auto.direct on /usr/contrib/games type autofs ignore,direct
  /etc/auto.direct on /opt/tools type autofs ignore,direct
  /etc/auto.direct on /var/mail type autofs ignore,direct
```


© 2003 Hewlett-Packard Development Company, L.P.

At this point, **games**, **tools**, and **mail** haven't been mounted yet. However, AutoFS does display the mount points for these file systems:

```
# ll -d /usr/contrib/games /opt/tools /var/mail
  dr-xr-xr-x   3 root sys  1024 Mar 28 08:50  /usr/contrib/games
  dr-xr-xr-x   3 root sys  1024 Mar 28 08:50  /opt/tools
  dr-xr-xr-x   3 root sys  1024 Mar 28 08:50  /var/mail
```

The first time a user accesses one of the directories managed by the direct map, AutoFS automatically mounts the file system associated with that directory:

```
# ll /usr/contrib/games
  -r-xr-xr-x   3 root sys  1024 Mar 28 08:50  tetris
  -r-xr-xr-x  44 root sys  1024 Mar 29 13:54  xpilot
  -r-xr-xr-x  18 root sys  1024 Mar 24 12:17  chess

# mount -v
  /etc/auto.direct on /usr/contrib/games type autofs ignore,direct
  /etc/auto.direct on /opt/tools type autofs ignore,direct
  /etc/auto.direct on /var/mail type autofs ignore,direct
  gamesvr:/usr/contrib/games on /usr/contrib/games type nfs
     ro,rsize=32768,wsize=32768,NFSv3
```

Configuring the AutoFS Direct Map

In order to configure a direct map, verify that **/etc/auto_master** contains a direct map entry. The first field of the direct map entry in **/etc/auto_master** must be "**/-**". The second field specifies the full pathname for the direct map file itself. You may change the direct map filename if you wish.

```
# vi /etc/auto_master
  /- /etc/auto.direct
```

Next, create the **/etc/auto.direct** file. Each entry in the direct map has three fields:

- The first field identifies the full pathname of a mount point directory that AutoFS should monitor.
- The second field lists the mount options AutoFS should use when mounting the file system. This field is optional.
- The third field identifies the file system to mount on the mount point identified in the first field.

In order to mount **/usr/contrib/games**, **/opt/tools**, and **/var/mail** via AutoFS, the following entries would be required in **/etc/auto.direct**:

```
# vi /etc/auto.direct
  /usr/contrib/games  -ro  gamesvr:/usr/contrib/games
  /opt/tools          -ro  toolsvr:/opt/tools
  /var/mail           -rw  mailsvr:/var/mail
```


© 2003 Hewlett-Packard Development Company, L.P.

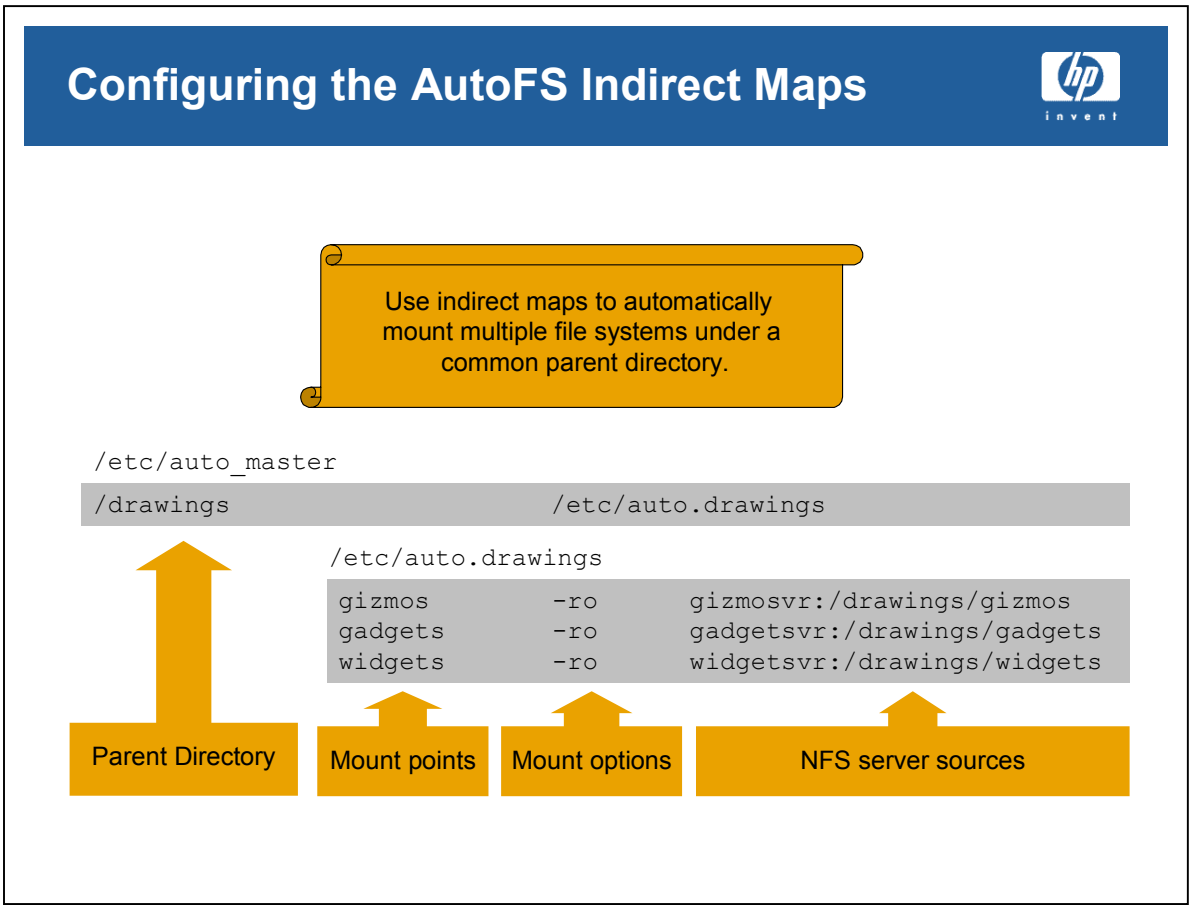
Finally, execute **/usr/sbin/automount** to make the changes take effect:

```
# /usr/sbin/automount
```

NOTE: Be sure to execute **/usr/sbin/automount** to update the mount table anytime you update the direct map file.

© 2003 Hewlett-Packard Development Company, L.P.

10–8. SLIDE: Configuring AutoFS Indirect Maps



Student Notes

An indirect map proves useful when you want AutoFS to mount several NFS file systems under a common parent directory.

The sample **`/etc/auto.drawings`** file on the slide automatically:

- Mounts **`/drawings/gizmos`**, read-only, from the **`gizmosvr`**
- Mounts **`/drawings/gadgets`**, read-only, from the **`gadgetsvr`**
- Mounts **`/drawings/widgets`**, read-only, from the **`widgetsvr`**

Example

If the **`/etc/auto_master`** and **`/etc/auto.drawings`** are configured as shown on the slide, you would see the following entry in your client's mount table initially. (Note that local file systems and the mount time stamps have been omitted for the sake of clarity.)

```
# mount -v
  /etc/auto.drawings on /drawings type autofs ignore,indirect
```


© 2003 Hewlett-Packard Development Company, L.P.

At this point, none of the drawing file systems have been mounted yet. In fact, the mount points have not even been created yet! Users that list the contents of the **/drawings** directory may be somewhat perplexed by the fact that the directory appears to be empty!

```
# ll /drawings
  total 0
```

The first time a user accesses one of the directories managed by the indirect map, AutoFS creates the necessary mount point directory and mounts the associated file system.

```
# ll /drawings/gizmos
  -r-xr-xr-x   3 root sys  1023 Mar 30 08:50  gizmo1
  -r-xr-xr-x  44 root sys   405 Mar 30 13:54  gizmo2
  -r-xr-xr-x  18 root sys   789 Mar 30 12:17  gizmo3

# mount -v
  /etc/auto.drawings on /drawings type autofs ignore,indirect
  gizmosvr:/drawings/gizmos on /drawings/gizmos type nfs
     ro,rsize=32768,wsize=32768,NFSv3
```

The other file systems under **/drawings** will only be mounted as needed.

Configuring the AutoFS Indirect Map

In order to configure an indirect map, you must first add an entry to **/etc/auto_master**. The first field in the indirect map **/etc/auto_master** entry identifies the full pathname for the parent directory under which AutoFS will mount the indirect map's file systems. The second field specifies the full pathname for the indirect map file. If your system uses multiple indirect maps, you may have multiple indirect map entries in **/etc/auto_master**.

```
# vi /etc/auto_master
  /drawings /etc/auto.drawings
```

As always, you must execute **/usr/sbin/automount** anytime you modify **/etc/auto_master**:

```
# /usr/sbin/automount
```

Next, create the indirect map **/etc/auto.drawings** file. Each entry in the indirect map has three fields:

- The first field identifies the relative pathname of a mount point directory that AutoFS should monitor.
- The second field lists the mount options AutoFS should use when mounting the file system. This field is optional.
- The third field identifies the file system to mount on the mount point identified in the first field.

© 2003 Hewlett-Packard Development Company, L.P.

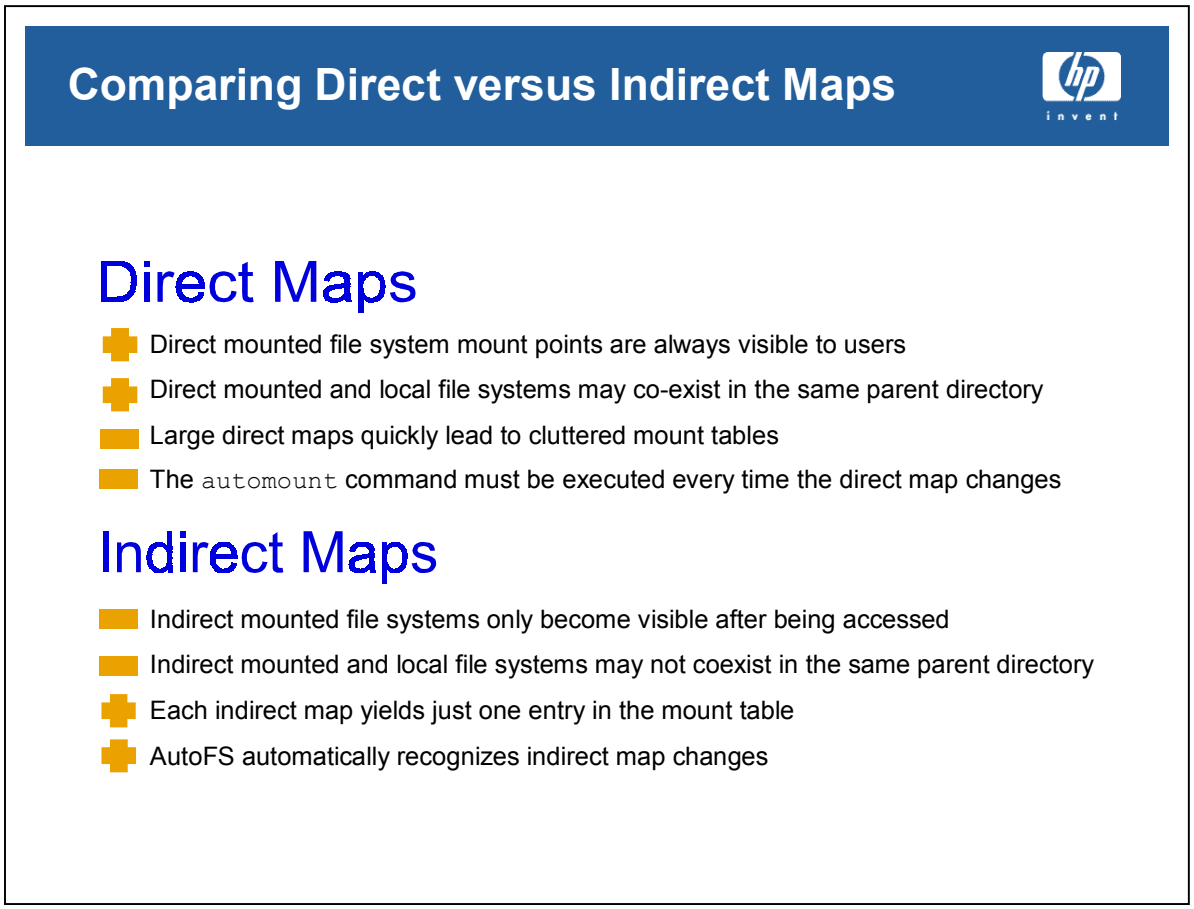
In order to mount **/drawings/gizmos**, **/drawings/gadgets**, and **/drawings/widgets** via AutoFS, the following entries would be required in **/etc/auto.drawings**:

```
# vi /etc/auto.drawings
  gizmos   -ro  gizmosvr:/drawings/gizmos
  gadgets  -ro  gadgetsvr:/drawings/gadgets
  widgets  -ro  widgetsvr:/drawings/widgets
```

NOTE:	You must execute **/usr/sbin/automount** anytime you change an indirect map entry in **/etc/auto_master**. However, it is not necessary to execute the **automount** command if the contents of the indirect maps themselves change.

© 2003 Hewlett-Packard Development Company, L.P.

10–9. SLIDE: Comparing Direct versus Indirect Maps



Student Notes

Determining when to use direct versus indirect maps is one of the most confusing issues faced by AutoFS administrators. The slide above and table below compare and contrast these two different AutoFS map types. The table references the sample direct and indirect maps shown below:

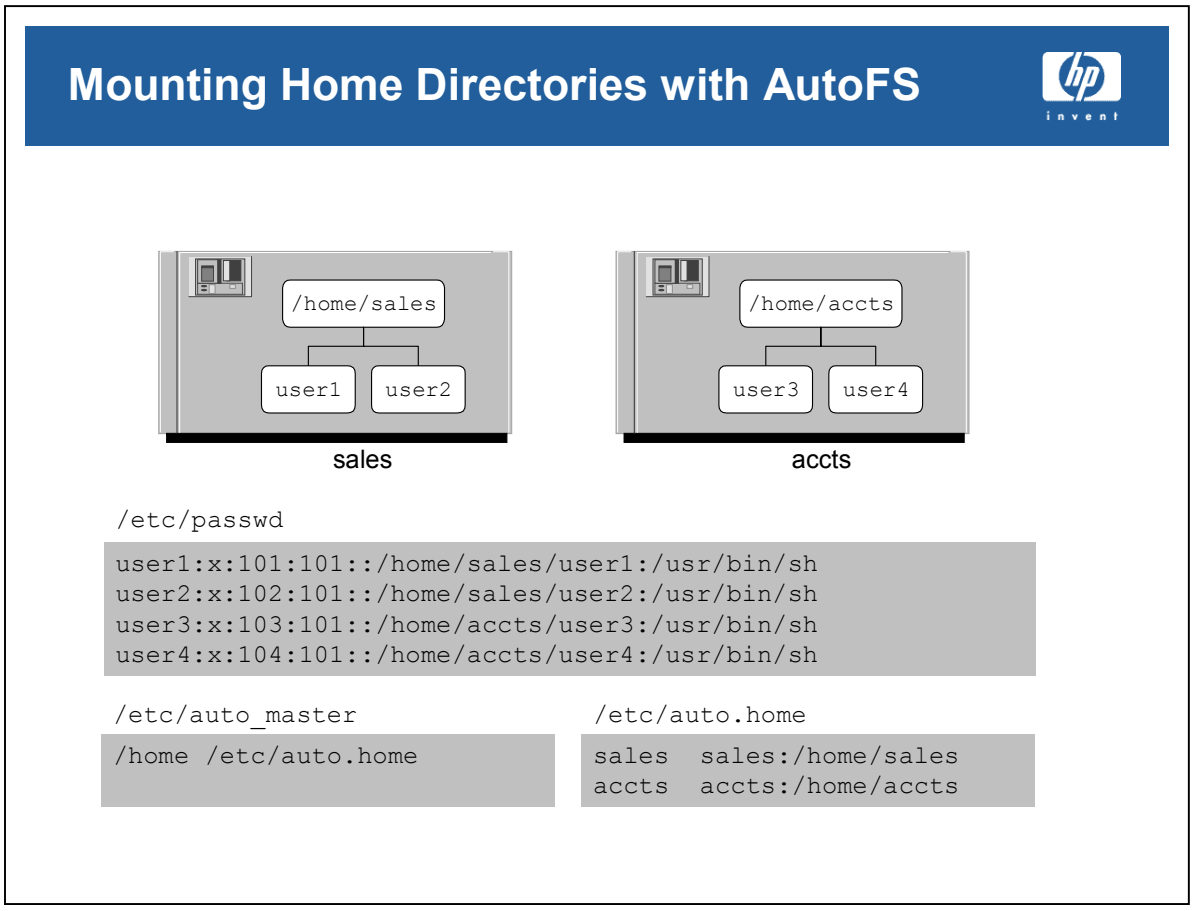
```
# cat /etc/auto_master
  /hosts                    -hosts -soft,nosuid
  /drawings                 /etc/auto.drawings
  /-                        /etc/auto.direct

# cat /etc/auto.direct
  /usr/contrib/games  -ro   gamesvr:/usr/contrib/games
  /opt/tools          -ro   toolsvr:/opt/tools
  /var/mail           -rw   mailsvr:/var/mail

# cat /etc/auto.drawings
  gizmos              -ro   gizmosvr:/drawings/gizmos
  gadgets             -ro   gadgetsvr:/drawings/gadgets
  widgets             -ro   widgetsvr:/drawings/widgets
```


© 2003 Hewlett-Packard Development Company, L.P.

Direct Maps	Indirect Maps
Advantage: Direct mounted file systems are always visible to users. If a system were configured with the sample direct map shown above, users could view the contents of **/usr/contrib** or **/usr/contrib/games** at any time.	Disadvantage: Indirect mounted file systems only become visible after being accessed. If a system were configured with the indirect map shown above, the **/drawings** directory would appear empty unless the user explicitly accessed a subdirectory within **/drawings**.
Advantage: Direct mounted AutoFS file systems and local file systems may coexist in the same parent directory. For example, the **/usr/contrib** directory on the sample system above contains both locally stored directories (such as **/usr/contrib/bin**) and an AutoFS direct map file system (**/usr/contrib/games**).	Disadvantage: Indirect mounted and local file systems may not co-exist in the same parent directory. For example, files stored locally under the **/drawings** directory on the sample system above would be hidden by the **/etc/auto.drawings** indirect map.
Disadvantage: Large direct maps quickly lead to cluttered mount tables. Each entry added to the direct map adds an entry to the mount table, too. Thus, the sample system shown above would have three AutoFS entries in the mount table as a result of the direct map.	Advantage: Each indirect map yields just one entry in the mount table. The sample indirect map shown above would create one mount table entry for **/drawings**.
Disadvantage: The **automount** command must be executed every time the direct map changes.	Advantage: AutoFS automatically recognizes indirect map changes. If you modify a directory's entry in an indirect map, AutoFS will see the changes the next time it mounts the directory; there is no need to execute the **automount** command.

© 2003 Hewlett-Packard Development Company, L.P.

10–10. SLIDE: Mounting Home Directories with AutoFS



Student Notes

User home directories are among the most commonly exported directories in NFS environments. If all of your home directories are on a single NFS server, then it might make sense for clients to mount **/home** from the server via an entry in **/etc/fstab**. NFS mounting home directories via **/etc/fstab** becomes more complicated, however, if your home directories are stored on multiple NFS servers across your local area network. If your home directories are scattered across multiple NFS servers, use AutoFS!

Consider the example on the slide. This organization has two NFS home directory servers. The “sales” server stores home directories for all members of the “sales” department, and the “accts” server stores home directories for all members of the “accts” department. The following configuration greatly simplifies home directory management in this type of environment. Better yet, it guarantees that any user may log onto any AutoFS client and have access to their home directory!

1. On each NFS server, create a subdirectory under **/home** that matches the server’s host name. On host “sales” create a directory called **/home/sales**. On host “accts,” create a directory called **/home/accts**.

   If you are migrating existing systems to NFS mounted home directories, you may need to

© 2003 Hewlett-Packard Development Company, L.P.

move users' home directories from the clients' local disks to the new NFS servers.

```
sales# mkdir /home/sales
accts# mkdir /home/accts
```

2. Create a home directory for each user on the appropriate server.

```
sales# mkdir /home/sales/user1
sales# mkdir /home/sales/user2
accts# mkdir /home/accts/user3
accts# mkdir /home/accts/user4
```

3. Export the **/home** file system on both servers.

```
sales# exportfs –i /home
accts# exportfs –i /home
```

4. Create an indirect map entry in **/etc/auto_master** to handle all attempts to access directories under **/home**. For the sake of clarity, name the map **/etc/auto.home**:

```
clients# vi /etc/auto_master
          /home /etc/auto.home
```

5. Create the **/etc/auto.home** map. Create one entry in the map for each server that exports home directories. For instance, the "sales" home directories should be mounted from **sales:/home/sales**. The "accts" home directories should be mounted from **accts:/home/accts**.

```
clients# vi /etc/auto.home
            sales  sales:/home/sales
            accts  accts:/home/accts
```

6. Update the home directory pathnames in the clients' **/etc/passwd** files. The home directory pathnames must be updated to reflect the new **/home/*servername*/*username*** directory naming convention. Note that *all* of the clients' **/etc/passwd** files must be updated.

```
clients# usermod –d /home/sales/user1 user1
clients# usermod –d /home/sales/user2 user2
clients# usermod –d /home/accts/user3 user3
clients# usermod –d /home/accts/user4 user4
```


© 2003 Hewlett-Packard Development Company, L.P.

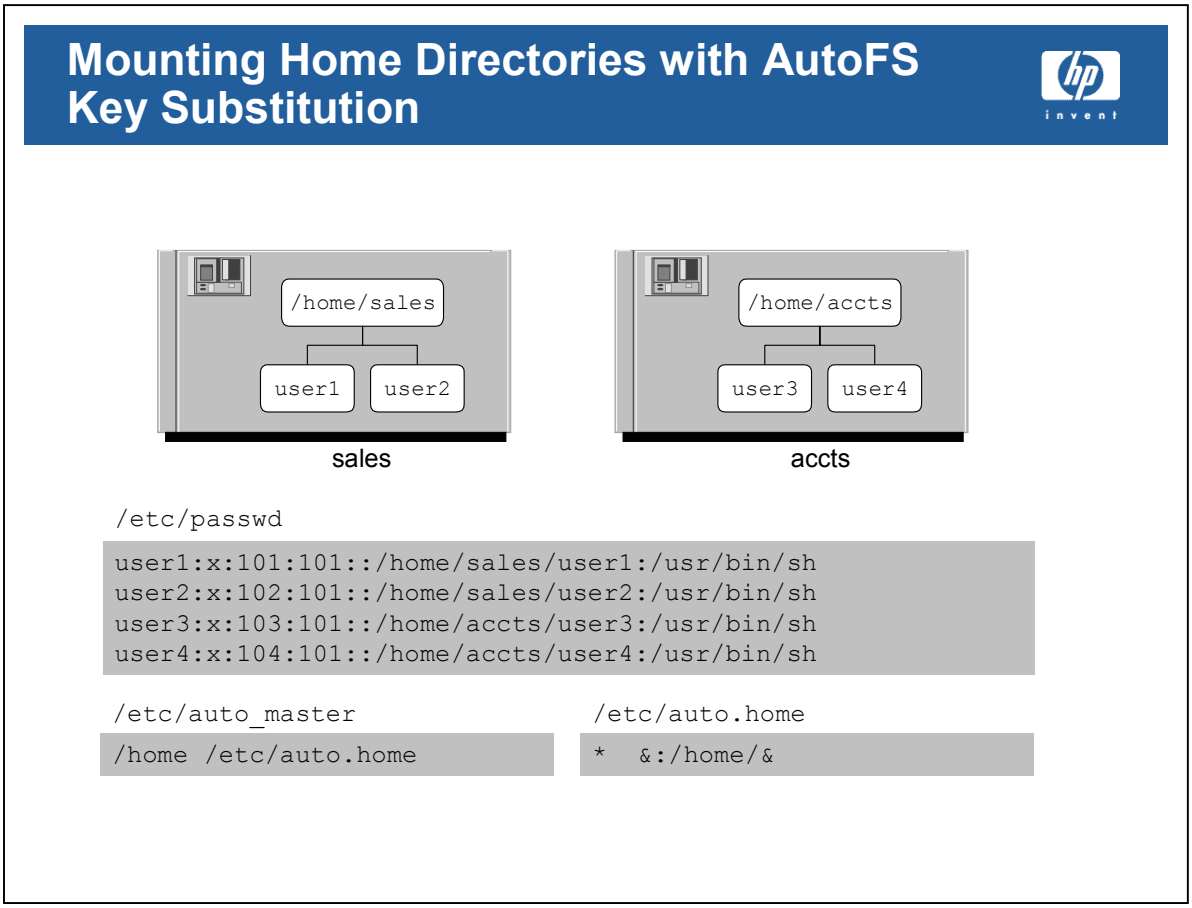
Questions

1. What type of map is being used in the example on the slide to automatically mount user home directories?

2. Why is this type of map preferable to its alternative? (Hint: What must be done each time a client's direct map file changes?)

© 2003 Hewlett-Packard Development Company, L.P.

10–11. SLIDE: Mounting Home Directories with AutoFS Key Substitution



Student Notes

The previous slide showed how AutoFS indirect maps can be used to automatically mount user home directories. The example on the slide showed a simple **/etc/auto_home** file that included references to just two NFS home directory servers:

```
clients# cat /etc/auto.home
         sales  sales:/home/sales
         accts  accts:/home/accts
```

With just two NFS servers, the **/etc/auto.home** file is easy to manage. Larger organizations, however, oftentimes have complex **/etc/auto.home** files that reference four, eight, sixteen, or even more NFS servers. Worse yet, changes made to **/etc/auto.home** must be propagated out to every one of your NFS clients!

Fortunately, AutoFS **key substitution** can simplify the administrator's life considerably in large NFS environments by replacing references to specific servers and file systems with two special wild card characters.

© 2003 Hewlett-Packard Development Company, L.P.

The first of these special characters is the ampersand (**&**). Consider the improved **/etc/auto.home** file below:

```
clients# cat /etc/auto.home
         sales  &:/home/&
         accts  &:/home/&
```

Each **&** in the map will automatically be replaced by the key value shown in the first field of the AutoFS map entry. Thus, the ampersands in the first line will be replaced by "**sales**," and the ampersands in the second line will be replaced by "**accts**." This abbreviated map saves the NFS client administrator a few keystrokes, while still providing the same functionality as the **/etc/auto.home** map on the previous slide.

The map file may be further condensed to a single line by replacing the key field in **/etc/auto.home** with an "*****" wildcard. Assuming that **/etc/auto.home** is an AutoFS map mounted on **/home**, then any attempt to access anything under **/home** matches the "*****" entry.

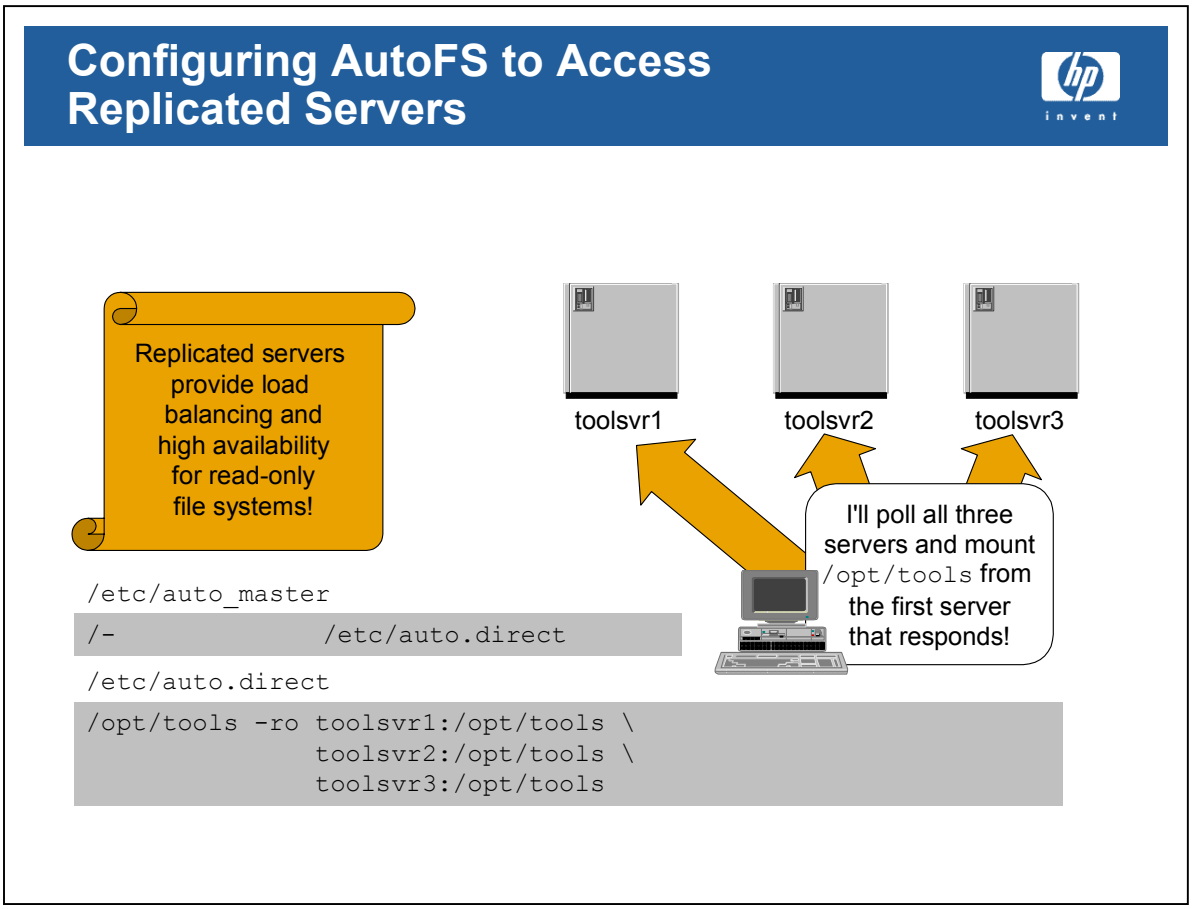
```
clients# cat /etc/auto.home
         * &:/home/&
```

Consider the following example: **user1** types **cd /home/sales/user1**. Since the **/etc/auto.home** map is mounted on **/home**, AutoFS intercepts the access attempt. AutoFS searches the **/etc/auto.home** map for a matching entry. Although the map never explicitly states which server should be used to mount the **sales** subdirectory, AutoFS does find the "*****" wildcard entry, which matches the key, **sales**. Using **sales** as the key value, AutoFS substitutes the ampersands on the right side of the map entry and mounts **sales:/home/sales**.

This simple, single-line map allows AutoFS to mount home directories from any NFS home directory server on the network. Furthermore, the administrator can add additional home directory servers to the environment without modifying AutoFS maps on the NFS clients.

© 2003 Hewlett-Packard Development Company, L.P.

10–12. SLIDE: Configuring AutoFS to Access Replicated Servers



Student Notes

All of the map files discussed in the chapter so far have listed exactly one NFS server for each AutoFS mount point. However, it turns out that the AutoFS direct and indirect maps can actually list two, three, or even more NFS servers for each AutoFS mount point. This **Replicated Server** functionality can dramatically improve performance for AutoFS clients that mount executables and other read-only file systems via AutoFS.

The example on the slide shows three NFS servers: toolsvr1, toolsvr2, and toolsvr3. All three servers have identical copies of the **`/opt/tools`** application directory, which is made available to clients via NFS.

Note that the direct map file responsible for mounting **`/opt/tools`** is a bit different than the maps discussed up to this point: instead of listing one server as a source for mounting **`/opt/tools`**, the map lists all three servers!

```
# cat /etc/auto.direct
  /opt/tools -ro toolsvr1:/opt/tools \
                 toolsvr2:/opt/tools \
                 toolsvr3:/opt/tools
```


© 2003 Hewlett-Packard Development Company, L.P.

This could also be written as follows:

```
# cat /etc/auto.direct
  /opt/tools -ro toolsvr1,toolsvr2,toolsvr3:/opt/tools
```

When a user accesses the **/opt/tools** directory, **automountd** polls all three servers and mounts the file system from the server that responds first. This functionality provides several advantages:

- Minimized network traffic. Since servers on the local network segment can respond more quickly to AutoFS client polls than servers on other segments, clients are more likely to choose a replicated server on the local network. This minimizes NFS traffic across your routers and gateways.

- Load balancing. Since heavily-loaded servers can't respond to client polls as quickly as lightly-loaded servers, new clients will likely choose to mount replicated file systems from the lightly-loaded servers.

- Reliability. Even if one of the NFS servers is down at the time of the request, the client will still be able to mount the file system from one of the other replicated servers. Note, however, that once AutoFS chooses a server, the selection is static. If a server becomes unavailable after a client has mounted a file system, **automountd** will not dynamically switch to one of the remaining servers.

CAUTION:	To ensure data consistency regardless of the NFS server chosen by the AutoFS client, the replicated server functionality should only be used for read-only file systems.

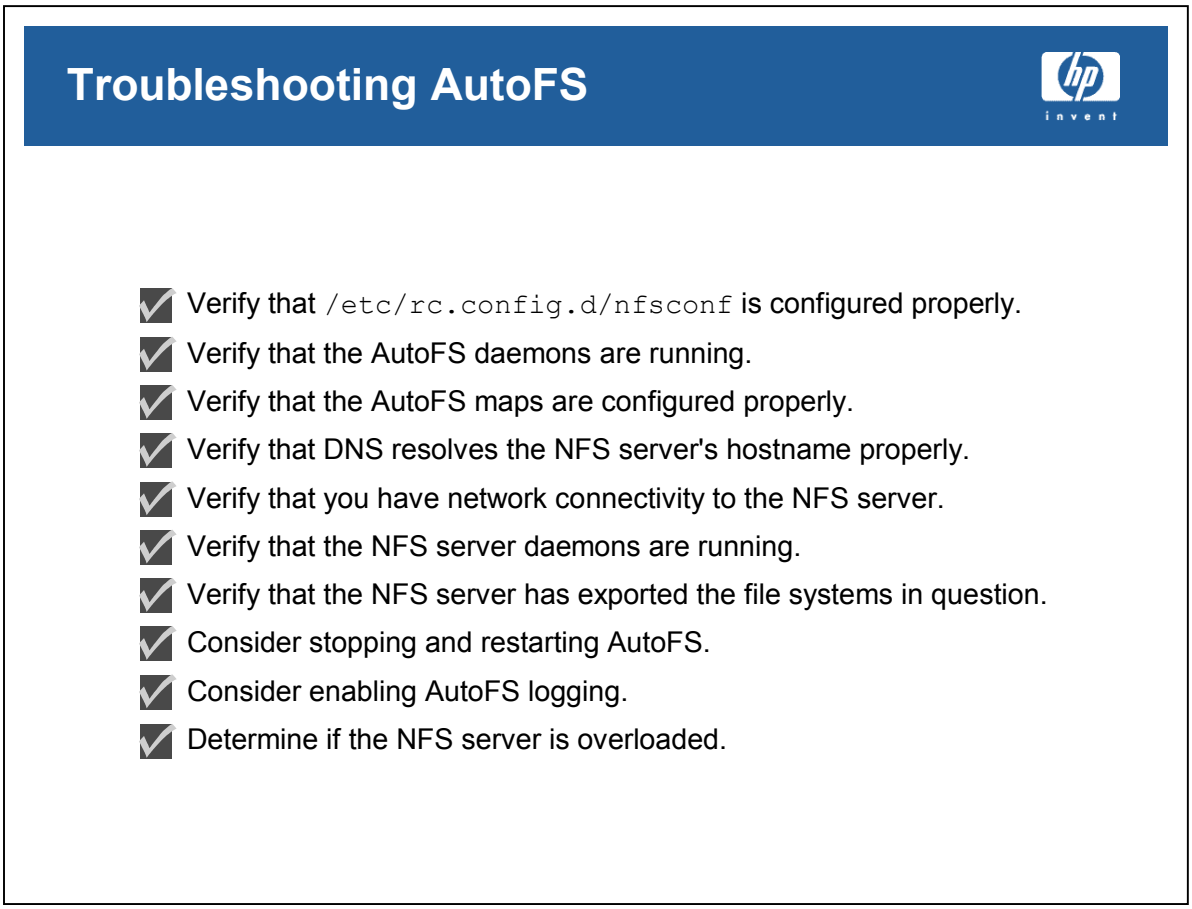
The configuration on the slide shows a very simple replicated server configuration. In more complex NFS environments, you can choose to assign weights to each replicated server. The lower a server's weight value, the more likely it is that that server will be chosen by AutoFS. Servers without an explicitly assigned weight value have a weight value of 0. In the example shown below, toolsvr1 takes precedence of toolsvr2, and toolsvr2 takes precedence over toolsvr3.

```
# cat /etc/auto.direct
  /opt/tools -ro toolsvr1(1):/opt/tools \
                 toolsvr2(2):/opt/tools \
                 toolsvr3(3):/opt/tools
```

Server proximity is more important than the weights you assign. A server on the same segment as the client is more likely to be selected than a server on the other side of a gateway, regardless of the assigned weights.

© 2003 Hewlett-Packard Development Company, L.P.

10–13. SLIDE: Troubleshooting AutoFS



Student Notes

If AutoFS appears to be misbehaving, try the following:

Verify that `/etc/rc.config.d/nfsconf` is Configured Properly

Check the **`nfsconf`** file to verify that the following variables are defined properly:

```
# cat /etc/rc.config.d/nfsconf
  NFS_CLIENT=1
  AUTOMOUNT=1
  AUTOFS=1
```

Verify that the AutoFS Daemons are Running

The AutoFS daemons must be running in order for AutoFS to function properly. Verify that this is the case by executing the **`ps`** command. If the daemons aren't running, re-run the **`nfs.client`** start script.

```
# ps -e | grep -e autofs_proc -e automountd
```


© 2003 Hewlett-Packard Development Company, L.P.

Verify that the AutoFS Maps are Configured Properly

Do all of the AutoFS maps appear in the mount table? If so, consult the map files themselves to check the mount options and NFS server names.

```
# mount -v | grep "type autofs"
# cat /etc/auto*
```

Verify that DNS Resolves the NFS Server's Host Name Properly

Since AutoFS maps reference NFS servers by host name, DNS problems can cause problems for AutoFS. Use **nsquery** to verify that your client is able to resolve each of the NFS server names to IP addresses.

```
# nsquery hosts server
```

Verify that you have Network Connectivity to the NFS Server

Are you able to **ping** the server? If you can't **ping** the server, AutoFS won't be able to send mount requests to the server. Check your IP address, your routing table, and your connectivity to other hosts on the network.

```
# ping server
```

Verify that the NFS Server Daemons are Running

Verify that **rpc.mountd** and **nfsd** are both registered with the NFS server's **rpcbind** daemon. If the server's NFS daemons aren't listed, ask the server administrator to re-run **/sbin/init.d/nfs.server start**.

```
# rpcinfo -u server mountd
# rpcinfo -u server nfs
```

Verify that the NFS Server has Exported the File Systems in Question

AutoFS can only mount file systems that have been exported by the NFS server. Use the **showmount** command to verify that the file systems you need have been properly exported.

```
# showmount -e server
```

Consider Stopping and Restarting AutoFS

If all else fails, consider stopping and restarting AutoFS.

```
# /sbin/init.d/nfs.client stop
# /sbin/init.d/nfs.client start
```

Does the startup script generate any error messages?
Can you start the service manually?

```
# /usr/lib/netsvc/fs/autofs/automountd
# /usr/sbin/automount
```


© 2003 Hewlett-Packard Development Company, L.P.

Consider Enabling AutoFS Logging

You might also consider enabling verbose AutoFS tracing and logging. With this functionality enabled, you will be able to determine exactly which mount requests are generated by AutoFS.

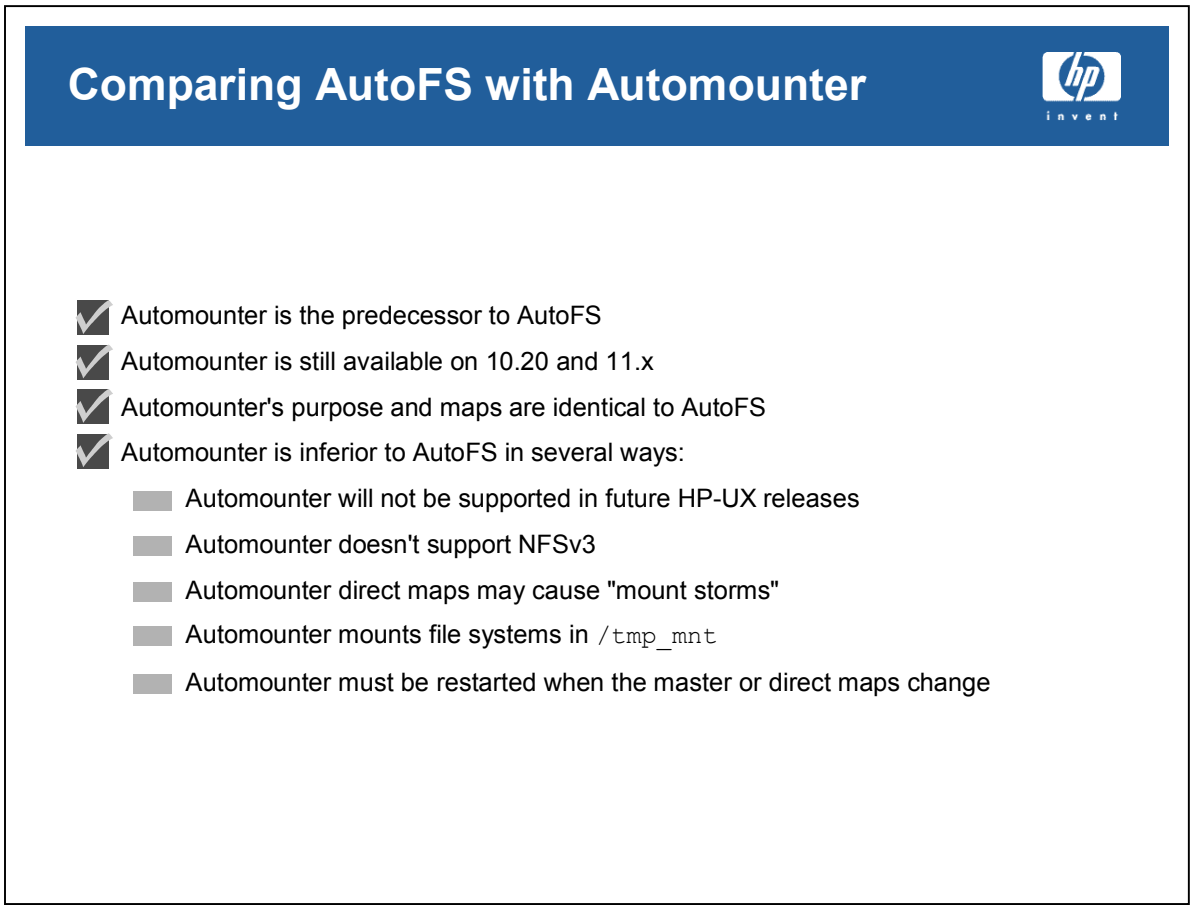
```
# vi /etc/rc.config.d/nfsconf
  AUTOMOUNT_OPTIONS="-v"
  AUTOMOUNTD_OPTIONS="-v -T"
# /sbin/init.d/nfs.client stop
# /sbin/init.d/nfs.client start
# more /var/adm/automount.log
```

Determine if the NFS Server is Overloaded

As far as NFS is concerned, a slow server is equivalent to a downed server. If your server is overloaded, your mount requests may timeout, and cause problems for AutoFS. Run **`glance`** or **`sar`** on the NFS server to determine if the server might be the problem.

© 2003 Hewlett-Packard Development Company, L.P.

10–14. SLIDE: Comparing AutoFS with Automounter



Student Notes

AutoFS has only been supported in HP-UX since 1998. Prior to the release of AutoFS, HP-UX provided similar functionality via the Automounter service. Automounter is still supported in HP-UX 10.20 and 11.x, but is quickly being supplanted by AutoFS for several reasons:

- Automounter will not be supported in future releases of HP-UX. Although both Automounter and AutoFS are supported in 10.20 and 11.x, HP has announced that future releases of the OS will not support the older Automounter service.

- Automounter doesn't support NFS Protocol Version 3. Protocol Version 3 introduced support for large files over 2 GB, and numerous performance enhancements. None of this new functionality is available to clients mounting file systems via the traditional Automounter and NFS Protocol Version 2.

- Automounter direct maps may cause "mount storms." If an Automounter direct map referenced several file mount points under a common parent directory, doing an **`ll`** on the parent directory caused all of the file systems below that directory to mount immediately – whether they were needed or not! This placed an unnecessary burden on the NFS servers. AutoFS direct maps don't cause mount storms.

© 2003 Hewlett-Packard Development Company, L.P.

- Automounter mounts file systems under **/tmp_mnt**. The traditional Automounter always mounted file systems under the **/tmp_mnt** directory, then used a complex web of symbolic links to make it appear as if the file systems were mounted in the normal **/usr**, **/opt**, **/home**, etc. file systems. This oftentimes confused users and administrators alike.

- Automounter must be stopped and restarted whenever **/etc/auto_master** or **/etc/auto.direct** change. There is no way to dynamically modify the master or direct maps when using the traditional Automounter service. In order to change these maps, the administrator must stop and restart the Automounter daemon. Unfortunately, in order to restart the Automounter daemon properly, you must first kill any processes using file systems mounted by the previous instance of the daemon. This oftentimes required a reboot any time the master or direct map changed. In today's 24x7 environments, these frequent reboots are unacceptable. After changing an AutoFS master or direct map, you can dynamically execute the **automount** command to make the changes take effect immediately.

AutoFS first became available in 10.20 as part of an Additional Core Enhancement (ACE) release in 1998. AutoFS was first released for 11.00 as part of a Software Extension Pack the same year. To determine if AutoFS is installed on your system, simply check for the existence of the **/usr/lib/netsvc/autofs/automountd** executable.

Automount and AutoFS can – and usually do -- *coexist* on a system simultaneously, but may not be *running* concurrently on the same system. To determine which daemon you are running, check the **/etc/rc.config.d/nfsconf** file. If the **AUTOFS** variable is set to "1", you are running AutoFS rather than the traditional Automounter.

Fortunately, transitioning from the traditional Automounter to the newer AutoFS is a simple procedure. See HP's *Installing and Administering NFS Services with 10.20 ACE and HWE* manual for details.

© 2003 Hewlett-Packard Development Company, L.P.

10–15. LAB: Configuring AutoFS

Preliminary

This lab assumes that the classroom has been configured with the 128.1.*.* IP addresses configured earlier in the course. The instructor station must be assigned IP address 128.1.0.1. Execute the following preliminary setup steps on both the student and instructor workstations in preparation for the lab:

```
# /labs/autofs.lab.setup.sh
```

These scripts added several entries to the **/etc/passwd** and **/etc/hosts** files on both the instructor and student workstations. When executed on the instructor station, the script also configures several additional IP addresses via IP multiplexing, and creates and exports several directories.

Part 1: Enabling and Starting AutoFS

Before you can configure the AutoFS maps, you must verify that NFS is installed, and the AutoFS daemons are running. That's the goal of this first portion of the lab!

1. Verify that the NFS product is installed on your system, and that the NFS client functionality is configured in **/etc/rc.config.d/nfsconf**.

2. AutoFS was not included in the NFS product that was initially shipped with 10.20 and 11.00. Verify that AutoFS is included in the version of the NFS product installed on your system by checking for the existence of the **/usr/lib/netsvc/fs/autofs** directory.

3. HP-UX 10.20 and 11.x support both AutoFS and the older Automounter. Is either of these services configured on your machine? Which one, if any?

© 2003 Hewlett-Packard Development Company, L.P.

4. Enable AutoFS in **/etc/rc.config.d/nfsconf**, but don't try to start the daemon yet.

5. Automount and AutoFS should never run concurrently on a system. Technically, you should be able to switch from one service to the other by tweaking the control variables in **/etc/rc.config.d/nfsconf**. Realistically speaking, however, it is often difficult to shut down automounter without rebooting since the daemon won't die until all of the automounted file systems are unmounted. The cleanest solution is to reboot. Make it so!

   ```
   # shutdown -ry 0
   ```

6. When your system comes back up again, verify that the AutoFS daemons are running.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring the AutoFS **-hosts** Map

The **-hosts** map provides a convenient mechanism for automatically mounting NFS file systems from any NFS server without modifying **/etc/fstab** or issuing the **mount** command. This portion of the lab walks you through the steps required to configure the **-hosts** map.

1. The **-hosts** entry is included in **/etc/auto_master** by default in HP-UX. Verify that the map has already been configured in your system's **/etc/auto_master** file.

2. Does the mount table reflect the fact that AutoFS is managing the **/net** mount point?

3. Test your **-hosts** map! What happens when you access **/net/corp**? Try it!

```
# ls /net/corp
```

4. What changed in the mount table?

5. Will AutoFS recognize a host referenced by IP address rather than name? Try it!

```
# ls /net/128.1.0.1
# mount -v
```

6. What happens if you attempt to access a non-existent host? Try it!

```
# ls /net/10.1.1.1
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring the AutoFS Direct Map

This part of the lab exercise gives you an opportunity to supplement your **-hosts** special map with a direct map file, too.

1. Add a direct map entry to **/etc/auto_master**. Name your direct map **/etc/auto.direct**.

2. Configure your direct map to automatically mount the **/usr/contrib/games** directory from the **corp** NFS server. Use the read-only mount option.

3. What must be done to make this change take effect? Make it so!

4. What appears in the mount table to indicate that AutoFS has recognized the new direct map?

5. Does the **games** mount point appear when you list the contents of **/usr/contrib**? Does listing the **/usr/contrib** directory cause AutoFS to mount the **games** file system from the NFS server?

```
# ls /usr/contrib
# mount -v.
```


© 2003 Hewlett-Packard Development Company, L.P.

6. **cd** to **/usr/contrib/games**, and list the contents. There should be an executable under **games** called **/usr/contrib/games/oneko/bin/X11/oneko**. Run the **oneko** executable, then check the mount table to see what changed.

```
# cd /usr/contrib/games
# ls
# /usr/contrib/games/oneko/bin/X11/oneko &
# mount –v.
```

7. Add another entry to your direct map to mount the **/data/contacts** directory from the corp NFS server. Users will need both read and write access to this file system. Don't execute the **automount** command yet.

8. What happens at this point if you attempt to do an **ls** of **/data/contacts**?

9. Do whatever is necessary to make the **/data/contacts** directory available on the client. Verify that your fix works.

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Configuring an AutoFS Indirect Map

Your organization has three departments, with home directories on three different NFS servers. Members of the finance department have their home directories on a server called "finance," members of the business department have their home directories on a server called "business", and members of sales have their home directories on a server called "sales". Your goal in this portion of the lab exercise is to configure an indirect map that will mount and unmount these home directories on an as-needed basis.

1. The indirect map used in this portion of the lab will be mounted under **/home**. This will not work if the logical volume containing your current users home directories is also mounted on **/home**. For the remainder of this lab, unmount the logical volume containing your users' home directories.

   ```
   # umount /home
   ```

2. Add an indirect map entry for **/home** to **/etc/auto_master**. This map entry should reference the **/etc/auto.home** map file.

3. What must be done anytime the master map changes? Make it so!

4. Now create the **/etc/auto.home** map file. The map file should configured such that:

/home/finance	is mounted from	**finance:/home/finance**
/home/business	is mounted from	**business:/home/business**
/home/sales	is mounted from	**sales:/home/sales**

   Is it necessary to re-issue the **automount** command after creating/changing the indirect map file?

© 2003 Hewlett-Packard Development Company, L.P.

5. Check the mount table. How many mount table entries were created as a result of the new indirect map? How many entries would have been created in the mount table if this had been configured as a direct map?

6. Do an **`ls`** of **`/home`**. Can you explain the result? Did AutoFS mount any file systems?

7. Now access a specific user's home directory and see what happens to the mount table:

```
# ls /home/finance/user1
# mount -v
```

8. Will this configuration automatically mount a user's home directory at login time? Try it! Try logging in as user "user3." Then check the mount table to verify that the user's home directory was in fact mounted from the proper location.

```
# su - user3
$ pwd
$ ls -a
$ exit
# mount -v
```

9. Can you shorten the **`/etc/auto.home`** file to a single line? How? Make it so! Then test your solution:

```
# vi /etc/auto.home
# ls /home/sales/user5
# mount -v
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Cleanup

Before moving on to the next chapter, run the **`netfiles.sh`** cleanup script:

```
# /sbin/init.d/nfs.client stop
# mount -a
# /labs/netfiles.sh –r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 11 — Configuring NIS

Objectives

Upon completion of this module, you will be able to do the following:

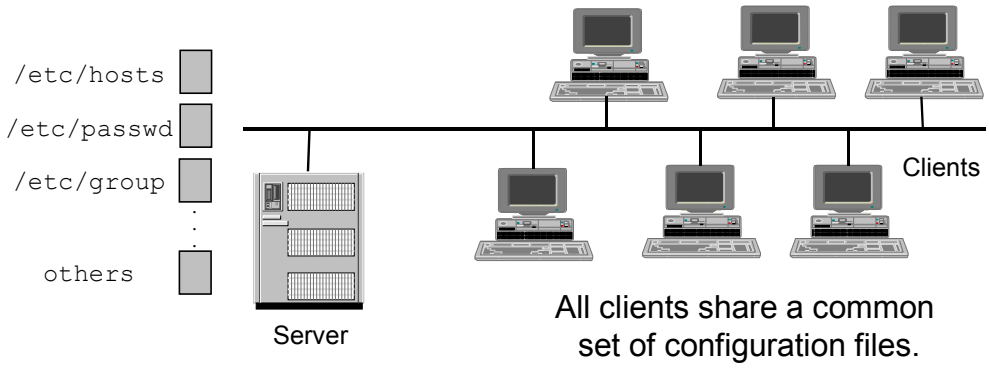
- Describe the purpose of Network Information Service (NIS).
- List the standard NIS maps.
- Configure an NIS master server.
- Configure an NIS slave server.
- Configure an NIS client.
- Change a password stored in the password map.
- Update other NIS maps on the master server.
- Propagate new maps to a slave server.
- Restrict user access to the master server.
- Describe the differences between NIS and NIS+.

© 2003 Hewlett-Packard Development Company, L.P.

11–1. SLIDE: Why Use NIS?

Why Use NIS?

- NIS provides for single point administration of system configuration files.
- NIS ensures consistency of files across the LAN.
- Files maintained by NIS include:



Student Notes

Every UNIX-based node on a network requires a certain amount of maintenance in order to stay current and up-to-date. For example, if a new node is added to the network, every UNIX-based system should have its **`/etc/hosts`** file updated to contain the name of the new node. Or, if a new user is added, and the user requires potential access to all nodes, every system will need its **`/etc/passwd`** file updated. With a few systems to update, this may seem reasonable. As the number of nodes increases however, the administration for these types of updates becomes time consuming and tedious.

Rather than manage the host names and user accounts on each individual system, a software tool called Network Information Service (NIS) was developed by Sun Microsystems to allow these files to be maintained on a single system (an NIS server) and referenced by other systems configured as NIS clients. With NIS, when a new host is added to the network, a single system's files are updated and these changes are propagated out to the other nodes on the network.

Another major advantage of NIS (besides central administration), is consistency across all nodes on the network. Because all systems reference the same set of files (referred to an NIS database files), users do not have to worry about which systems have which login accounts setup, or if they will be able to reference a new node by its host name on all machines.

© 2003 Hewlett-Packard Development Company, L.P.

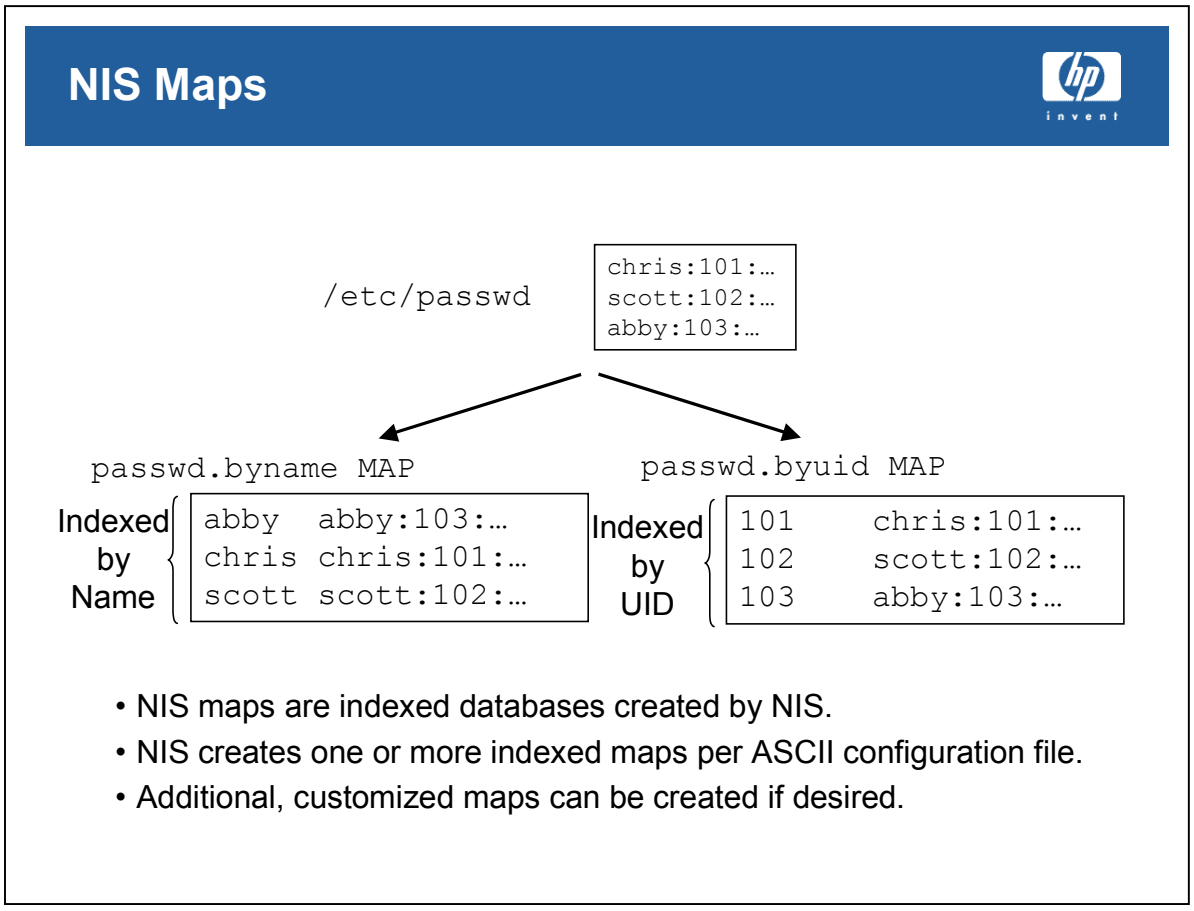
Preserving consistency means that if the information is available on one machine, it is available (with the exact same definition) on all machines on the network (using NIS).

In HP-UX, the NIS software is bundled with the NFS product and the default operating system.

NIS was formerly known as the Yellow Pages. However, this name is a registered trademark of British Telecommunications in the United Kingdom, so the name of the service was changed.

© 2003 Hewlett-Packard Development Company, L.P.

11–2. SLIDE: NIS Maps



Student Notes

The ASCII files that NIS uses are converted into databases files (also known as NIS map files) when NIS is configured. Each NIS map file is sorted based on common fields used to index into the file. For example, the **`/etc/passwd`** file is translated into NIS maps which index based on login names (**`passwd.byname`**), and based on UIDs (**`passwd.byuid`**).

There is one NIS map called **`ypservers`** that is not built from an ASCII source file. It is created automatically during NIS configuration. It contains a list of the master and slave servers for the NIS domain.

Each of the maps is appended with two suffixes when created: **`.pag`** and **`.dir`**. For example:

```
passwd.byname.dir
passwd.byname.pag
passwd.byuid.dir
passwd.byuid.pag
```

If your file system only supports short file names, a file name can only have 14 characters. This means map names can only be 10 characters in length because the **`.dir`** and **`.pag`** suffixes are added to map names. NIS will then create short map names:

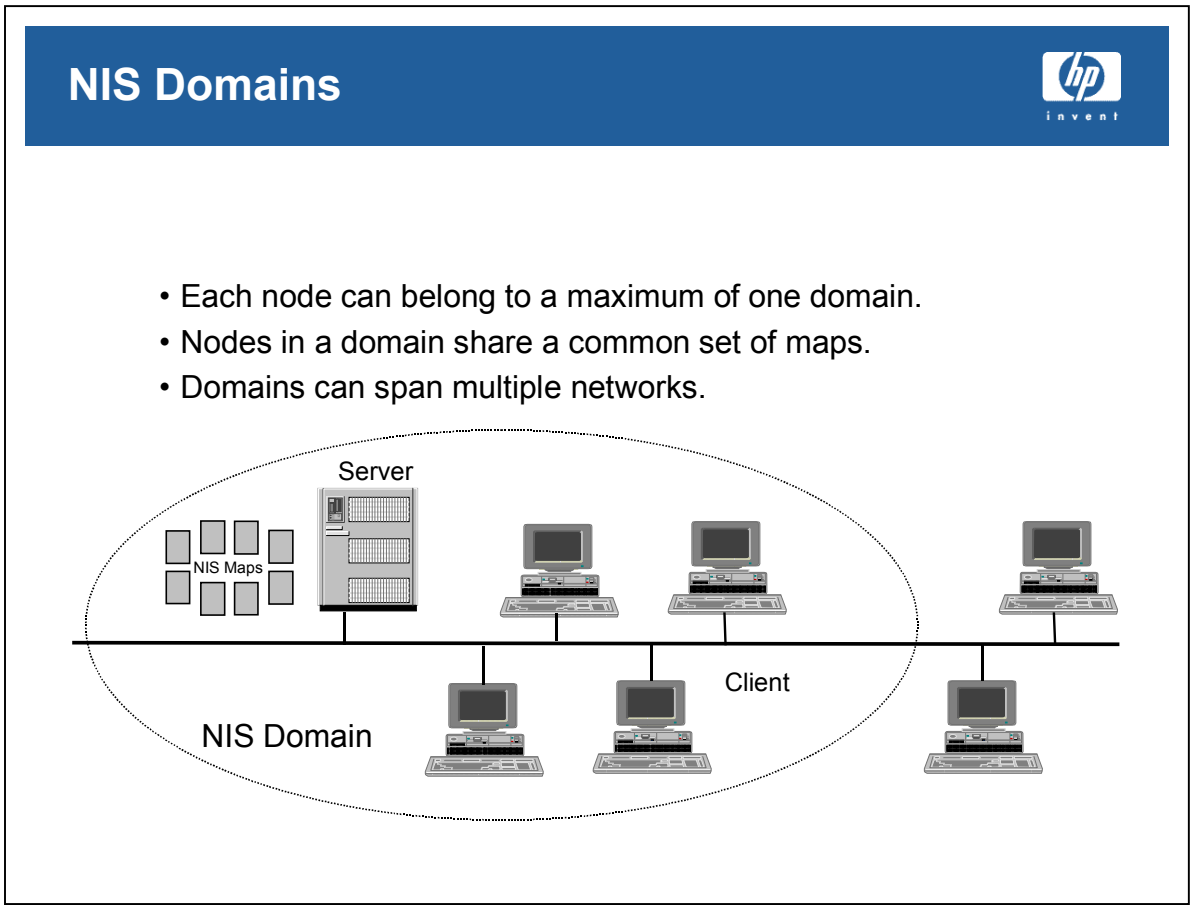
© 2003 Hewlett-Packard Development Company, L.P.

```
passw.byna.dir
passw.byna.pag
passw.byui.dir
passw.byui.pag
```

NOTE: An NIS map is synonymous with an NIS database file.

© 2003 Hewlett-Packard Development Company, L.P.

11–3. SLIDE: NIS Domains



Student Notes

An NIS domain is a logical grouping of nodes using the same NIS maps. There can be more than one NIS domain within a physical network. Nodes that have the same domain name belong to the same NIS domain.

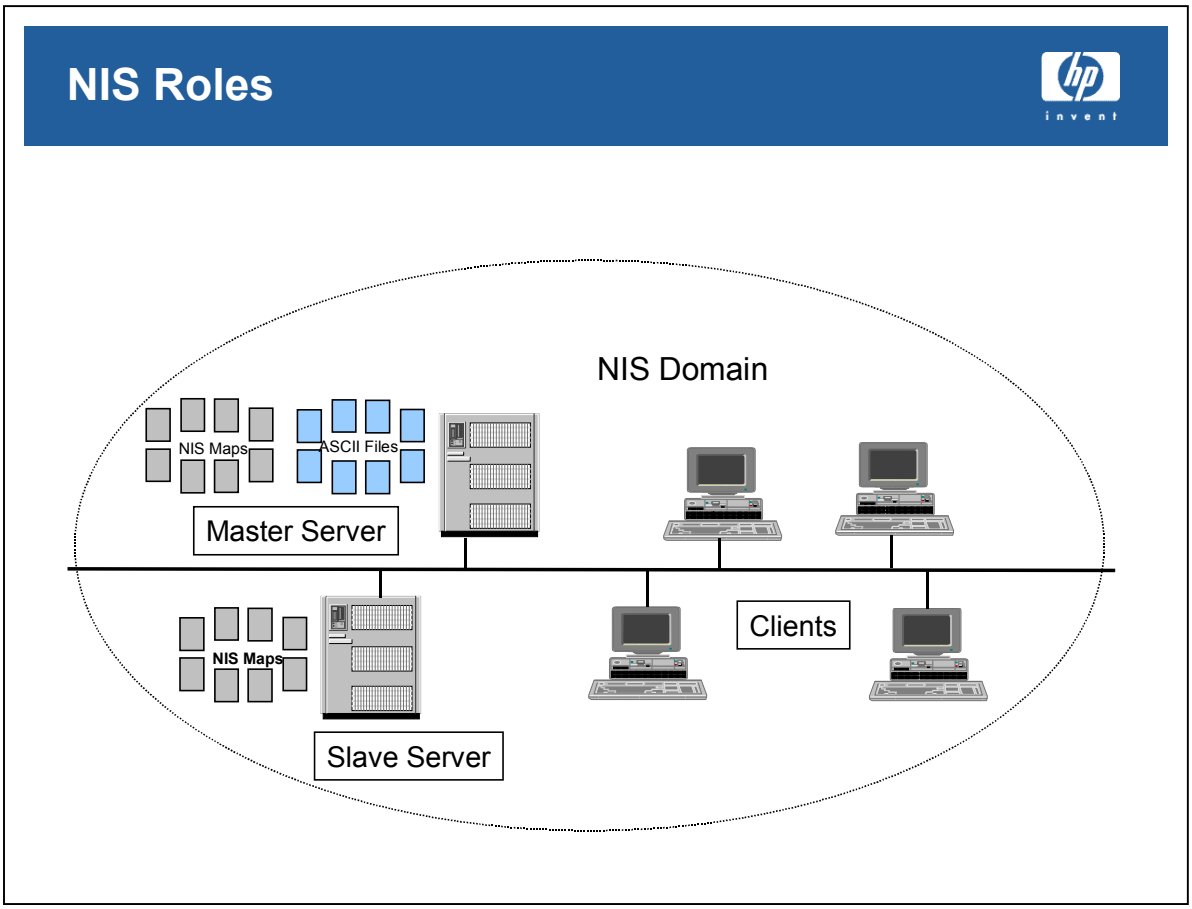
NIS domain-related files are stored under a subdirectory beneath **`/var/yp`** on the NIS servers. The subdirectory name corresponds to the name of the NIS domain which that system serves. For example, maps in the **`research`** domain would be stored in directory **`/var/yp/research`**. NIS domain names are case-sensitive. The NIS standard for systems supporting long file names is a domain name of up to 64 characters.

The **`/etc/rc.config.d/namesvrs`** file on each system has an **`NIS_DOMAIN`** variable, which is used to set the domain name for a system during boot configuration. The domain name may be changed interactively by the superuser by executing the **`/usr/bin/domainname`** command. Users can determine the default domain name on the local system by executing **`domainname`** with no parameters.

NOTE:	There is no relationship between NIS maps and DNS maps.

© 2003 Hewlett-Packard Development Company, L.P.

11–4. SLIDE: NIS Roles



Student Notes

The major components of a NIS domain include the master server, the slave servers, and the clients.

The **master server** is the system on which the original ASCII files are kept and modified. These files are translated into maps on the master server.

Slave servers have copies of the maps and, along with the master, serve the information over the network to the clients. Slave servers are optional.

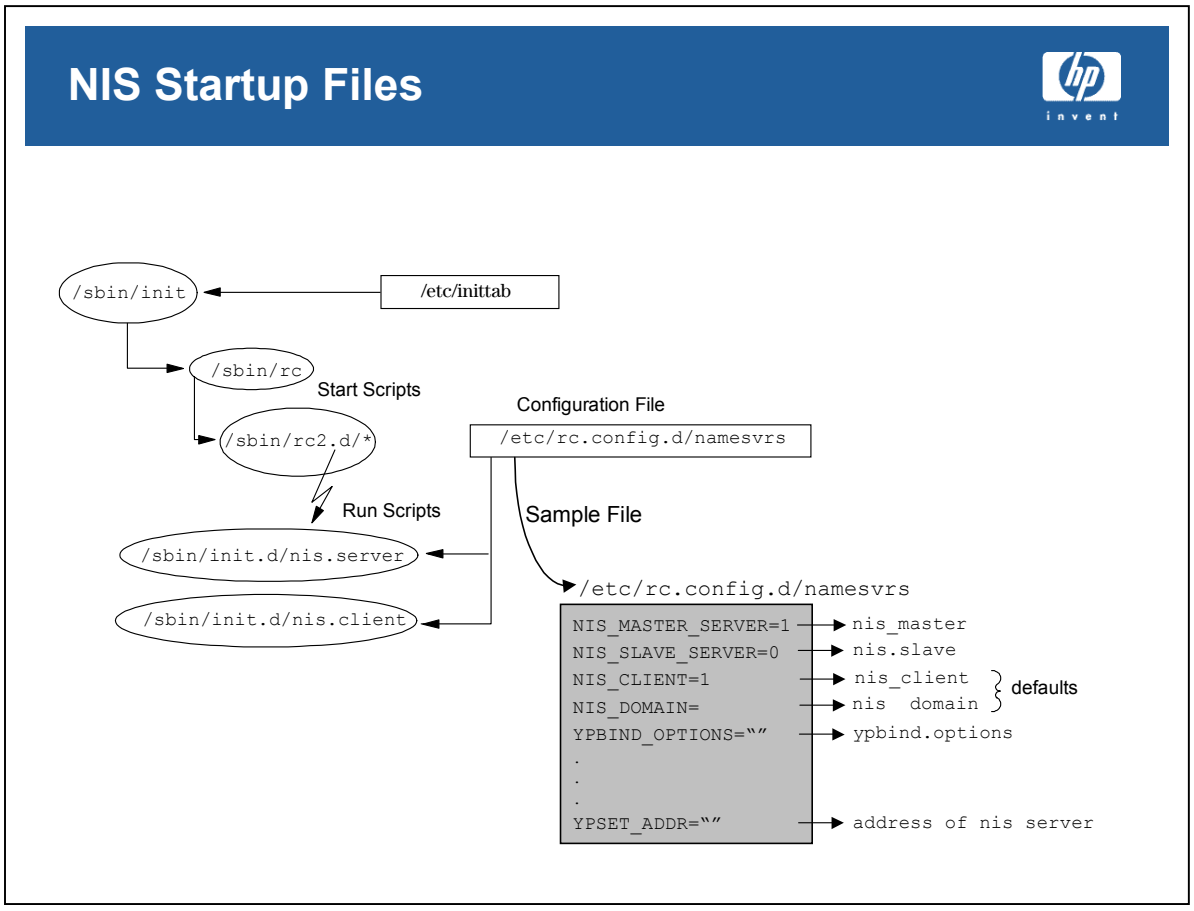
Clients do not have maps or copies of the server's ASCII files (though having their own local ASCII files as backups is desirable). They look up entries across the network from either the master or slave servers.

NIS servers and clients are different from NFS servers and clients.

- NIS servers provide access to information in NIS maps to NIS clients.
- NFS servers provide access to the server's file systems to NFS clients.
- While some systems may perform multiple NIS and NFS roles, there is no requirement that the systems be the same.

© 2003 Hewlett-Packard Development Company, L.P.

11–5. SLIDE: NIS Startup Files



Student Notes

When the system starts to run level 2 or higher, the start scripts (linked scripts) in **`/sbin/rc2.d`** will be executed to start NIS server and NIS client functionality.

The start scripts are linked to the run scripts that reside in **`/sbin/init.d`**. These scripts fetch the configurable parameters from the configuration file **`/etc/rc.config.d/namesvrs`**, but the daemons will only be invoked if the appropriate variables are set to the correct values.

Master and slave servers use the same technique to access the NIS maps like clients; therefore, both run scripts are executed when a NIS server system boots.

NIS run scripts are invoked before NFS client and server functionality is started.

The process **`init`** controls run levels of an HP-UX system. Its configuration file is **`/etc/inittab`**. The first entry in this file defines the default run level of a system.

© 2003 Hewlett-Packard Development Company, L.P.

The following table shows you which daemons and commands are invoked by the run scripts:

Table 1

Run Script	Daemons Started	Comments

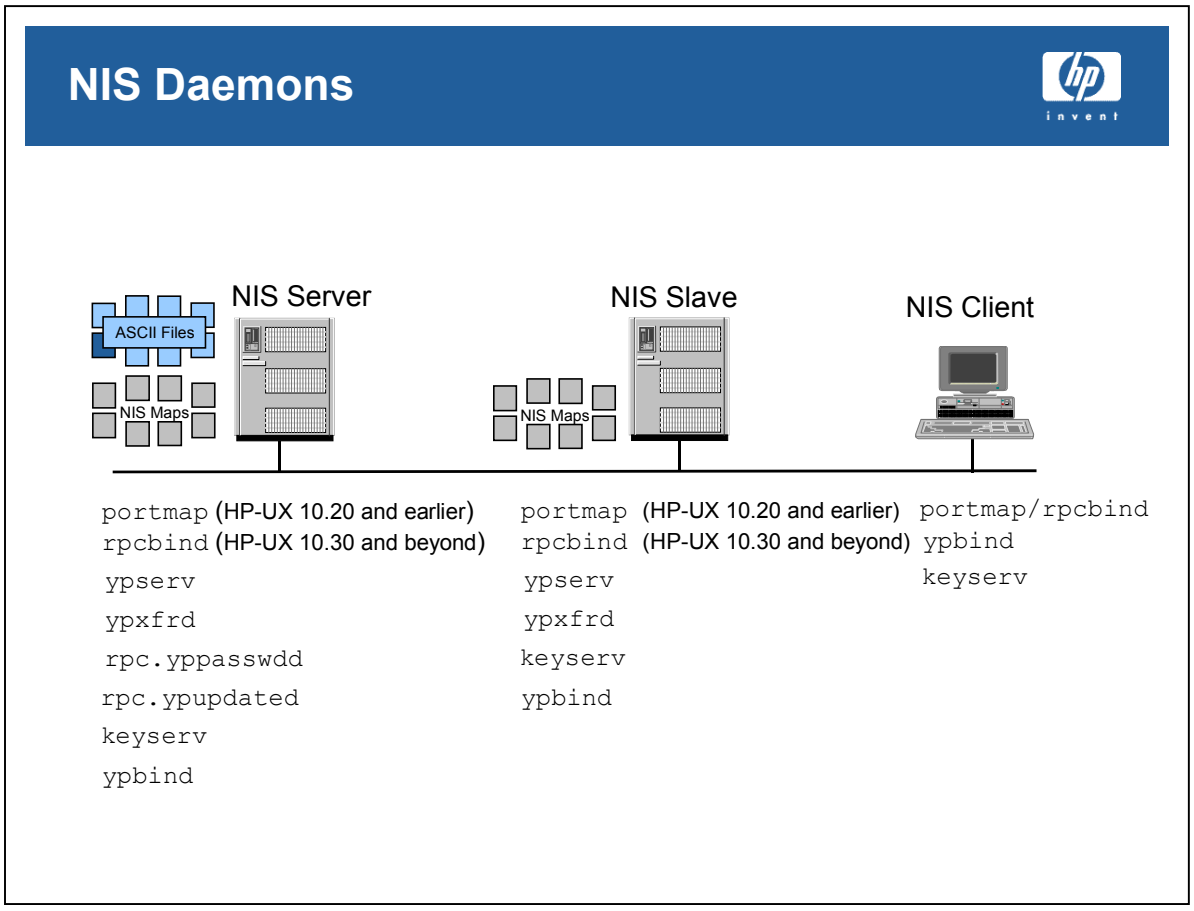
`nfs.core`	`portmap`
	HP-UX 10.20 and prior releases)
	`rpcbind`
	(HP-UX 10.30 and later releases)
		Only started if not already running

`nis.server`	`ypserv`
	`rpc.yppasswdd`
	`ypxfrd`
	`rpc.updated`
	`keyserv`
		NIS Server daemon
		Controls password file
		Transfers NIS maps
		For updating maps
		For secure RPCs

`nis.client`	`ypbind` `keyserv`	For binding to a server For secure RPCs

© 2003 Hewlett-Packard Development Company, L.P.

11–6. SLIDE: NIS Daemons



Student Notes

Several daemons associated with NIS follow.

NIS Master Server Only

`rpc.yppasswdd` The NIS passwd daemon (**`/usr/lib/netsvc/yp/rpc.yppasswdd`**) handles all password change requests from the **`yppasswd`** and **`passwd`** user commands. It changes passwords in the source file associated with the password map, rebuilds the map, and transfers it to all slave servers automatically.

`rpc.ypupdated` The **`rpc.ypupdated`** daemon provides a secure mechanism via secure RPCs, for updating an NIS map's source file on the NIS master, and regenerating the appropriate maps. This daemon is part of the secure RPC programming enhancement.

© 2003 Hewlett-Packard Development Company, L.P.

NIS Master Server and Slave Servers Only

ypserv The NIS database lookup server (**/usr/lib/netsvc/yp/ypserv**) looks up information in the local collection of maps in response to requests from clients.

ypxfrd A new daemon with HP-UX 10.0, **ypxfrd**, provides faster transfer of maps between master and slave servers.

All NIS Servers and All Clients

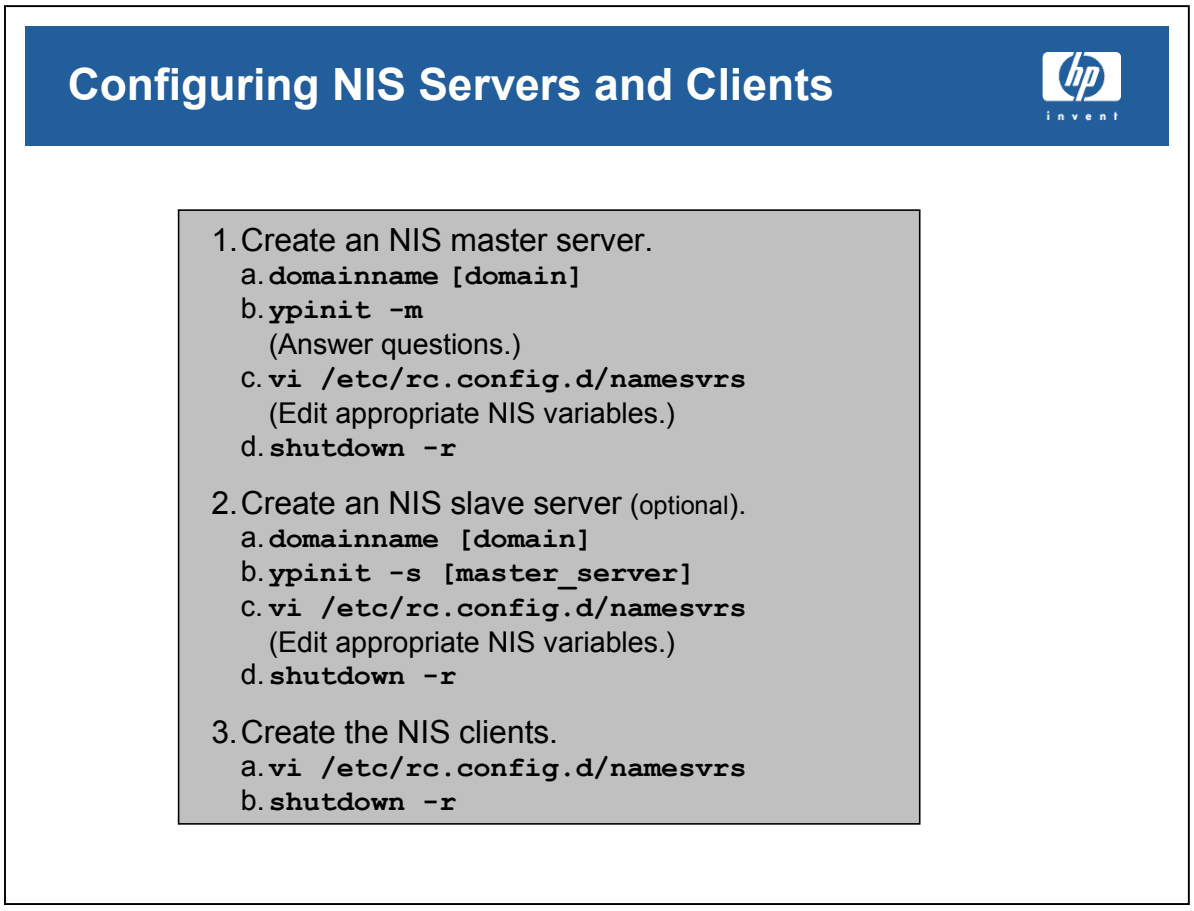
ypbind The NIS binder (**/usr/lib/netsvc/ypbind**) remembers information that lets client processes on the local machine communicate with **ypserv** processes.

keyserv **keyserv** stores the private encryption keys of all users logged into the system. This daemon is part of the secure RPC programming enhancement, and is not needed to access NIS maps.

NOTE: **portmapper** was replaced with **rpcbind** in release HP-UX 10.30.

© 2003 Hewlett-Packard Development Company, L.P.

11–7. SLIDE: Configuring NIS Servers and Clients



Student Notes

Now that you understand the major concepts surrounding NIS, we will show you how to configure NIS. The major steps are shown on the slide. We will discuss each step individually.

NOTE:	When you are creating a slave server, the maps are copied from the master server. Therefore, you must create the master server first.

Configuring an NIS Master Server

Below are the steps to configure an NIS master server:

1. Add the **/var/yp** directory to root's PATH variable. It contains the **ypmake** command to update maps.

2. Collect the ASCII source files, which are used to build the maps. They should be up to date.

3. Manually set the domain name.

```
# domainname research
```


© 2003 Hewlett-Packard Development Company, L.P.

4. Build and install databases.

```
# ypinit -m
---- supply slave server names interactively ----
```

5. Edit **/etc/rc.config.d/namesvrs**.

```
NIS_MASTER_SERVER=1
NIS_CLIENT=1
NIS_DOMAIN=research
```

6. Reboot.

```
# shutdown -ry 0
```

Configuring an NIS Slave Server

After the master server is configured, you can begin configuring the slave server:

1. Manually set the domain name.

```
# domainname research
```

2. Copy databases from the master.

```
# /usr/sbin/ypinit -s master_server
```

3. Edit **/etc/rc.config.d/namesvrs**.

```
NIS_SLAVE_SERVER=1
NIS_CLIENT=1
NIS_DOMAIN=research
```

4. Reboot.

```
shutdown -ry 0
```

Configuring an NIS Client

The following are the steps to configure an NIS client:

1. Edit **/etc/rc.config.d/namesvrs**.

```
NIS_CLIENT=1
NIS_DOMAIN=research
```

2. Ensure that at least one server is booted, then reboot your system.

```
shutdown -ry 0
```

After configuring the NIS master server, clients and slaves can be configured in any order.

© 2003 Hewlett-Packard Development Company, L.P.

11–8. SLIDE: Testing NIS

Testing NIS

- Are the server's daemons running?
  `# rpcinfo -p servername`
- Are the server's map files configured properly?
  `# yppoll -h servername -d domain passwd.byname`
- What domain am I a member of?
  `# domainname`
- Which server am I bound to?
  `# ypwhich`
- Which users are listed in the passwd map?
  `# ypcat -k passwd.byname`
- Is user1 included in the passwd map?
  `# ypmatch user1 passwd.byname`

Student Notes

After configuring NIS, there are several tools you can use to test your new configuration.

`rpcinfo -p servername`

First, use the **`rpcinfo`** command to verify that your NIS server is running the appropriate daemons. NIS uses remote procedure calls, just like NFS. The **`rpcinfo`** command contacts the server's **`portmap/rpcbind`** daemon and reports the server's registered RPCs. Master servers should be running **`ypserv`**, **`ypxfrd`**, **`yppasswdd`**, and **`ypupdated`**. Slave servers should be running **`ypserv`** and **`ypxfrd`**. If any of these daemons is missing, check your server's configuration!

`yppoll -h servername -d domain passwd.byname`

Next, use the **`yppoll`** command to verify that the server's map files are configured properly. Use the **`-h`** option to specify the hostname of the server you wish to query and the **`-d`** option to identify the domain in which you are interested. If the server daemons are running, and the server has the map you are searching for, it will return the map's "order number."

© 2003 Hewlett-Packard Development Company, L.P.

Each NIS map has an "order number" associated with it. Each time the master server rebuilds a map, that map's order number is incremented. NIS slave servers use these order numbers to determine if their local copies of the map files are up to date. If NIS is functioning properly, the order numbers on the slaves' maps should always match the order numbers on the master's maps.

`domainname`

If **`rpcinfo`** and **`yppoll`** both suggest that your server is functioning properly, you can begin checking your client configuration. The **`domainname`** command will tell you to which domain your client currently belongs.

`ypwhich`

The **`ypwhich`** command queries the local **`ypbind`** daemon to determine to which NIS server you are currently bound.

`ypcat -k passwd.byname`

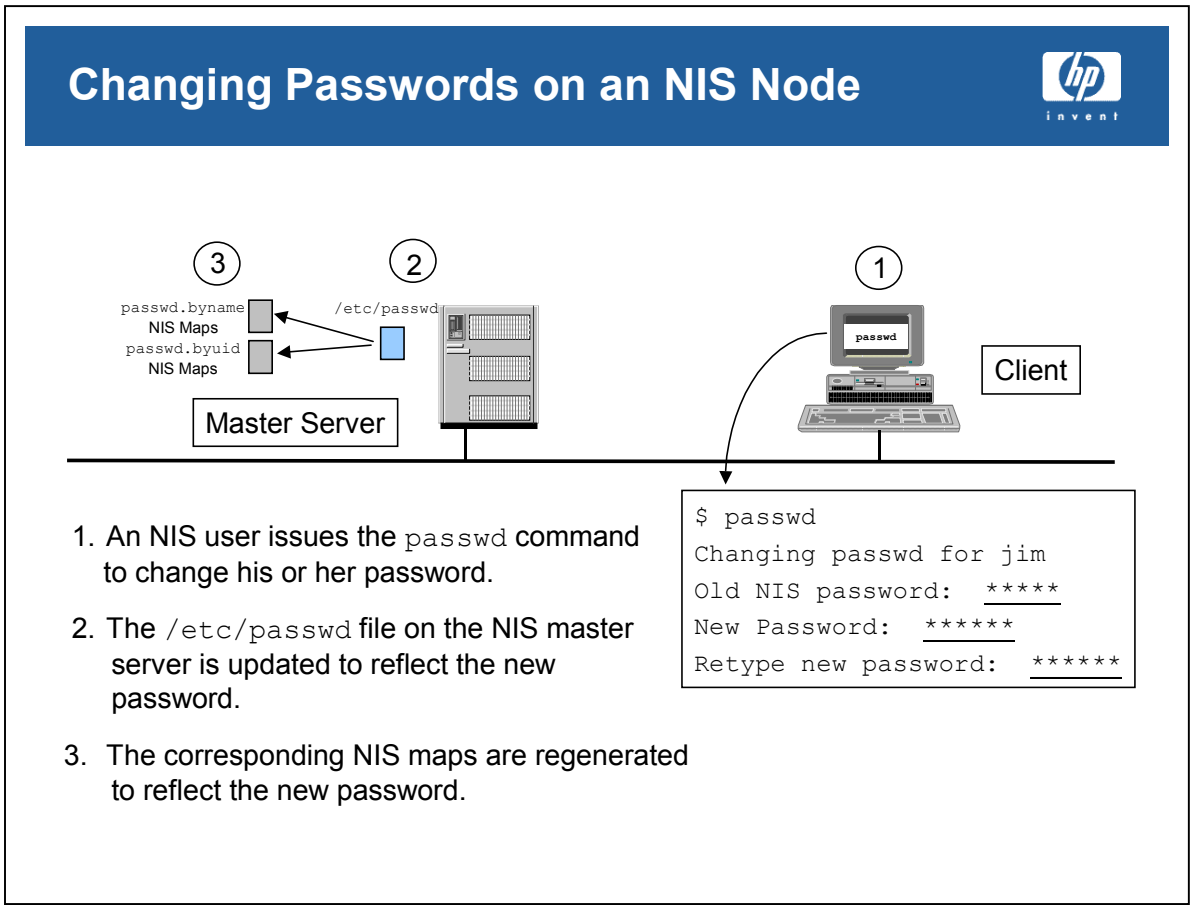
The **`ypcat`** command allows a client to dump the contents of an NIS server's maps. The **`-k`** option prepends the key value for each map entry on the beginning of each line.

`ypmatch user1 passwd.byname`

If you simply want to verify a single entry in an NIS map file, use **`ypmatch`**. The first **`ypmatch`** argument specifies the key value for which to search, and the second identifies the map you wish to search.

© 2003 Hewlett-Packard Development Company, L.P.

11–9. SLIDE: Changing Passwords on an NIS Node



Student Notes

If a user uses the **/usr/bin/passwd** command to change passwords, the login ID, old password, and new password are passed to the **rpc.yppasswdd** daemon of the NIS master server. After the old password is verified, **rpc.yppasswdd** updates the ASCII file and rebuilds the NIS maps (with **ypmake passwd**). Finally, the NIS slave servers receive a new copy of these maps, and the change is complete.

If a user is not administered by NIS (there is a complete local entry without escape character for this user), his or her password will be changed in the local **/etc/passwd**.

Prior to HP-UX 10.0, the user had to use the **yppasswd** command to change his or her password in an NIS environment. This command is still available, but you no longer need to use it.

Resetting Users' NIS Passwords

Occasionally, users forget their passwords. In a non-NIS environment, the system administrator can reset users' forgotten passwords by simply typing the **passwd** command with the user's *username* as an argument. Resetting user passwords in an NIS environment is a bit more complicated. The administrator must log in as root on the NIS master server,

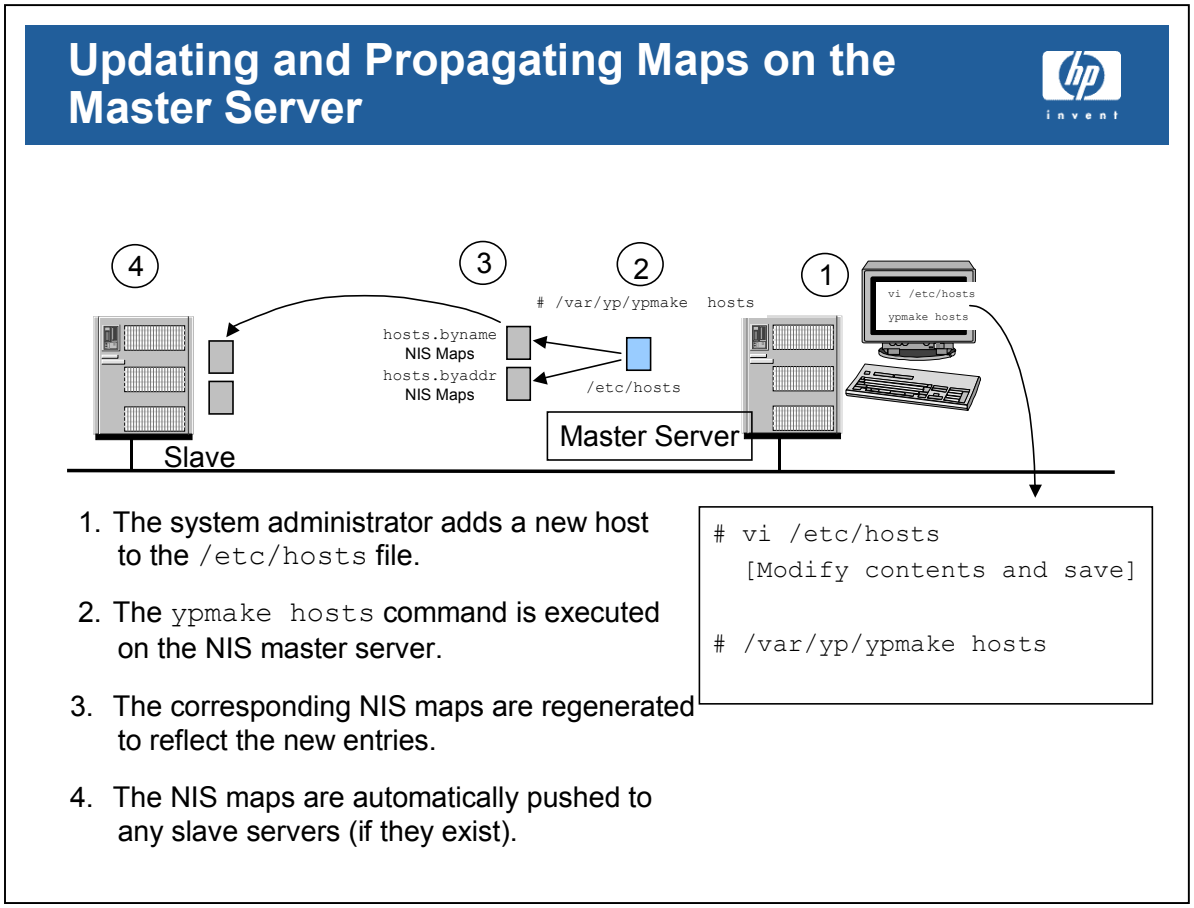
© 2003 Hewlett-Packard Development Company, L.P.

change the user's password in the master's **/etc/passwd** file, and update the NIS password map:

```
# passwd -r files username
# /var/yp/ypmake passwd
```


© 2003 Hewlett-Packard Development Company, L.P.

11–10. SLIDE: Updating and Propagating Maps on the Master Server



Student Notes

In order to update an NIS map, you must:

1. Modify the ASCII source file on the master server.
2. Rebuild the affected maps on the master server.
3. Push the updated maps out to the slaves.

There are several ways to update maps on the master server. The most straightforward way is to use the **ypmake** command. This command will take a source file, create a new map from it, and then "push" the new map to the slave servers. (It calls **yppush** to do this. We will talk about **yppush** in a moment.)

Example

```
# vi /etc/hosts
# /var/yp/ypmake hosts
```


© 2003 Hewlett-Packard Development Company, L.P.

```
For NIS domain research:

Building the hosts map(s)...  hosts build complete.
   Pushing the hosts map(s):  hosts.byaddr hosts.byname

ypmake complete: no errors encountered.
```

Another Example

```
# vi /etc/hosts
# /var/yp/ypmake
For NIS domain research:

The passwd map(s) are up-to-date.
The group map(s) are up-to-date.
Building the hosts map(s)...  hosts build complete.
   Pushing the hosts map(s):  hosts.byaddr hosts.byname
The networks map(s) are up-to-date.
        :
ypmake complete: no errors encountered.
```

`ypmake` Syntax

```
/var/yp/ypmake [DIR=path_to_source] \
               [DOM=NIS_domain] \
               [NOPUSH=num] \
               [PWFILE=passwd_file] [mapname]
```

The default *path_to_source* is **/etc**.

The **DOM** option lets you specify an NIS domain other than the host's default domain.

When not NULL, **NOPUSH** inhibits copying the new or updated databases to the slave NIS servers. (By default, the databases are copied to the slaves.) If you don't push the map (**ypmake NOPUSH=1 *mapname***), you can do it later with the **yppush** command.

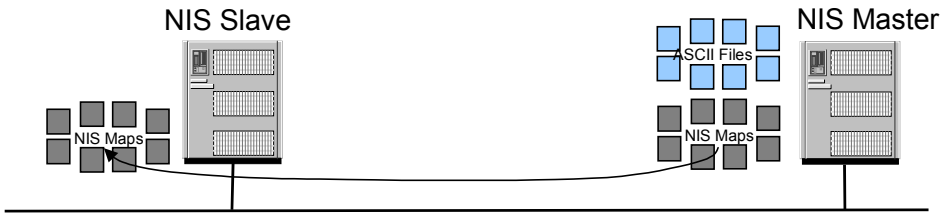
PWFILE allows the use of a password file other than **/etc/passwd**.

For more information, see **ypmake(1m)**.

© 2003 Hewlett-Packard Development Company, L.P.

11–11. SLIDE: Fetching Maps from the Master Server

Fetching Maps from the Master Server



- The `ypxfr` command
  - copies an NIS map from the master server to a slave
  - must be invoked on the slave server
  - transfers the map only if the master copy is more recent than the local copy

- The `ypxfr` command can be executed
  - interactively, running the command on the slave server
  - periodically, running the command from cron on each slave server
  - periodically, running the `yppush` command on the master server (`yppush` on the master server calls `ypxfr` on the slave)

Student Notes

When you set up NIS, **`ypinit`** copies maps from the master server to the slave servers. However, if you wish to keep the slave servers up-to-date, you should set up your system to periodically propagate maps to the slaves. This can be done with **`ypxfr`** in one of the following ways:

1. Periodically run **`ypxfr`** via **`cron`** on each slave server.

2. Interactively invoke **`ypxfr`** on a slave server.

```
# ypxfr passwd.byuid
ypxfr: map passwd.byuid at psd1 is not more recent than local
```

3. Periodically invoke **`yppush`** from the master server.

The **`ypxfr`** command uses the **`ypxfrd`** daemon to transfer the maps quickly. If **`ypxfrd`** daemon is not available, the transfer of the maps is done as in previous HP-UX releases.

© 2003 Hewlett-Packard Development Company, L.P.

Syntax

```
/usr/sbin/ypxfr [-h server] [-f] [-d domain] mapname
```

- Running **ypxfr** via cron allows you to execute **ypxfr** at different rates for different maps. For example, you could choose to update the **passwd** map once an hour and the **protocols** map once a day.

  The NIS service provides some scripts in the **/var/yp** directory that help you decide which maps should be updated hourly or daily. These scripts are called

  ypxfr_1perhour Fetches the maps

  ```
  passwd.byname,passwd.byuid
  ```

  ypxfr_1perday Fetches the maps

  ```
  group.bygid, group.byname, networks.byaddr
  networks.byname, protocols.byname
  protocols.bynumber, rpc.bynumber,
  services.byname ypservers, vhe_list
  ```

  ypxfr_2perday Fetches the maps

  ```
  ethers.byaddr, hosts.byaddr, hosts.byname
  mail.aliases, netgroup
  netgroup.byhost, netgroup.byuser
  ```

  You can use these scripts in conjunction with **cron** to update your maps. Your **crontab** entries could look something like the following:

  ```
  # At 11:30 am and 11:30 pm daily, transfer ethers, hosts,
  # mail.aliases and netgroup maps.
  30 11,23 * * * /var/yp/ypxfr_2perday
  #
  # At 45 minutes past the hour, transfer the passwd maps.
  45 * * * * /var/yp/ypxfr_1perhour
  ```

- You should only execute **ypxfr** interactively in exceptional situations. Testing a server and trying to solve a critical map inconsistency are good reasons. The following are the most frequently used options:

  -h *server* Allows you to get maps from servers other than the master server. This may come in handy if you are temporarily using another system as master or for testing.

  -d *domain* Allows you to copy a map from the domain specified (rather than the domain returned by **domainname**).

  -f Forces the map to be copied even if its order number at the remote NIS server is not more recent than the order number of the local map.

© 2003 Hewlett-Packard Development Company, L.P.

- You can also update NIS maps by executing **yppush** on the master server. **yppush** sends a transfer map request to each of the slave servers. In turn, **ypserv** on the server executes **ypxfr -C**. The **ypserv** daemon then passes **ypxfr** the information needed to identify and transfer the map. The syntax for **yppush** is

  ```
  /usr/sbin/yppush [-d domain] [-v] mapname
  ```

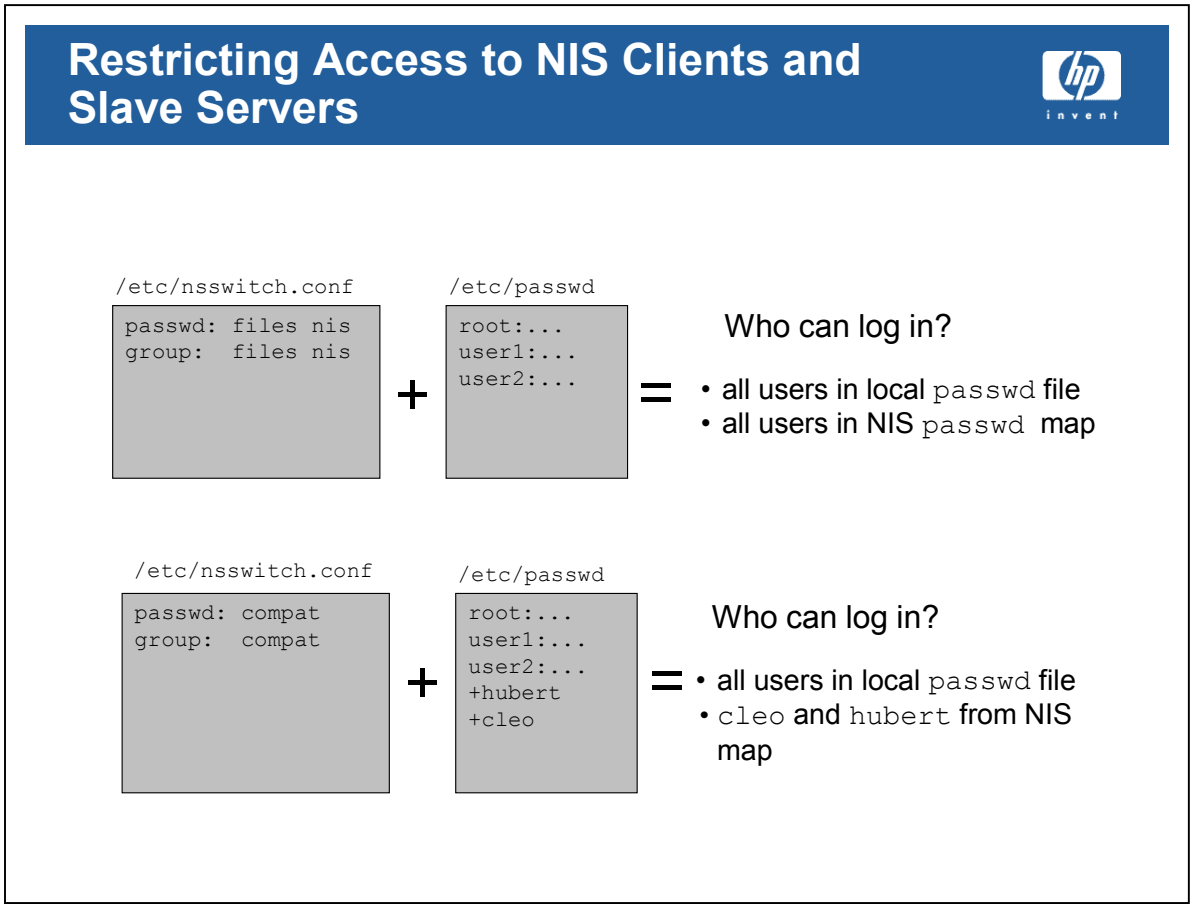
  For example,

  ```
  # yppush passwd.byname
  ```

For more information, see **yppush(1m)** and **ypxfr(1m)**.

© 2003 Hewlett-Packard Development Company, L.P.

11–12. SLIDE: Restricting Access to NIS Clients and Slave Servers



Student Notes

By default, when a user lookup is required, the system initially searches for the username in the local **/etc/passwd** file. If the username isn't found in **/etc/passwd** and NIS is configured, the system then consults the NIS **passwd** map. Using this approach, *all* the users both in the local password file *and* in the NIS map have access to all nodes in the NIS domain.

Many shops prefer to limit access to a given node to a more limited list of users. The **/etc/nsswitch.conf** file makes it possible to more narrowly define the concept of *if*, and *how*, a client uses the NIS maps. Each line in **/etc/nsswitch.conf** contains a type of lookup often performed by the system (for instance: **passwd**, **group**, **hosts**, and so forth), followed by a list of sources the system should consult when performing those lookups.

If a host should *only* use the *local* password and group files, and ignore the NIS **passwd** and **group** map, you should include the following lines in **/etc/nsswitch.conf**:

```
passwd:   files
group:    files
```


© 2003 Hewlett-Packard Development Company, L.P.

If, however, the host should allow *all* users defined either locally *or* in the NIS map to login, include the following two lines in **/etc/nsswitch.conf**. (Or, simply leave the **nsswitch.conf** file empty, as this is the default behavior anyway!)

```
passwd:  files nis
group:   files nis
```

If you want to allow *all* locally defined users, but only *selected* users from the NIS map to access a host, add the following two lines to **/etc/nsswitch.conf**:

```
passwd:  compat
group:   compat
```

After adding the compat entries, you will need to add *escape entries* to your **/etc/passwd** and **/etc/group** files to identify which NIS users should have access to the system.

The example below allows all locally defined users to access the system, as well as users **hubert** and **cleo** as defined in the NIS map. Other users defined in the NIS map will not have access to this system. Note the *escape entries* identified by the **+** signs. Allowing additional NIS users to access the system would simply require the addition of more escape entries.

```
root:ms0RtUNJemVSI:0:3::/:/sbin/sh
daemon:*:1:5::/:/sbin/sh
bin:*:2:2::/usr/bin:/sbin/sh
sys:*:3:3::/:
adm:*:4:4::/var/adm:/sbin/sh
uucp:*:5:3::/var/spool/uucppublic:/usr/lbin/uucp/uucico
lp:*:9:7::/var/spool/lp:/sbin/sh
nuucp:*:11:11::/var/spool/uucppublic:/usr/lbin/uucp/uucico
hpdb:*:27:1:ALLBASE:/:/sbin/sh
nobody:*:-2:60001::/:
+hubert
+cleo
```

Using escape entries in this manner allows the administrator to carefully control which users are allowed to login to each host in an NIS domain. Your database servers' **/etc/passwd** files, for instance, may only contain escape entries for the database administrators. Your accounting department workstations' **/etc/passwd** files may only contain escape entries for the users in the accounting department. Each administrator should carefully consider which users in the NIS map need access to each machine.

NOTE: The **compat** entry is mutually exclusive of any other value in the **passwd** field of the **/etc/nsswitch.conf** file.

We've only discussed the most common **nsswitch.conf** file possibilities here. The **nsswitch.conf man** page discusses the file format in detail. Several sample **nsswitch** files may be found in the **/etc** directory. Type **ls /etc/nsswitch.*** and copy the version

© 2003 Hewlett-Packard Development Company, L.P.

of the file that best meets your needs to **`/etc/nsswitch.conf`**, or simply leave the file empty or nonexistent if you want to allow all NIS users to log into your NIS client.

© 2003 Hewlett-Packard Development Company, L.P.

11–13. SLIDE: Restricting Access to the Master Server

Restricting Access to the Master Server

Use an alternate password file as the source for the password maps and reduce `/etc/passwd` on the master server.

1. Create an alternate password file as the source for the maps.
2. Reduce the `/etc/passwd` file and add escape entries.
3. Add `passwd:compat` and `group: compat` to `/etc/nsswitch.conf`.
4. Modify `YPPASSWDD_OPTIONS` in `/etc/rc.config.d/namesvrs`.
5. Stop and start NIS server functionality.
6. Modify the `PWFILE` variable in `/var/yp/ypmake`.
7. Modify the `PWFILE` variable in `/var/yp/Makefile`.
8. Rebuild and propagate the new password maps.

Student Notes

By default, the master server uses **`/etc/passwd`** as the map source file. If all home directories are available on the master server, all users can log into the master server. If you want to restrict access to a smaller set of users than defined by the complete **`/etc/passwd`**, perform the following steps on the master server:

1. Create an alternate password file as source for the maps.

   ```
   # cp /etc/passwd /etc/passwd.nis
   ```

2. Reduce **`/etc/passwd`** (remove users) and add escape entries.

   ```
   # vipw
   ```

3. NIS will not recognize your escape entries in the **`/etc/passwd`** file unless you add the following lines to your **`/etc/nsswitch.conf`** file:

   ```
   passwd: compat
   group:  compat
   ```


© 2003 Hewlett-Packard Development Company, L.P.

4. Edit **/etc/rc.config.d/namesvrs** and modify **YPPASSWDD_OPTIONS**.

   Change

   ```
   YPPASSWDD_OPTIONS="/etc/passwd -m passwd PWFILE=/etc/passwd"
   ```

   to

   ```
   YPPASSWDD_OPTIONS="/etc/passwd.nis -m passwd PWFILE=/etc/passwd.nis"
   ```

   This tells the **yppasswdd** daemon to manage **/etc/passwd.nis** instead of **/etc/passwd**. This change becomes active when **yppasswdd** is restarted.

5. Stop and activate NIS server functionality:

   ```
   # /sbin/init.d/nis.server stop
   # /sbin/init.d/nis.server start
   ```

6. Edit **/var/yp/ypmake** and modify **PWFILE**. Change the line

   ```
   PWFILE=${PWFILE:-$DIR/passwd}
   ```

   to

   ```
   PWFILE=${PWFILE:-$DIR/passwd.nis}
   ```

7. Edit the **/var/yp/Makefile** and modify **PWFILE**. Change

   ```
   PWFILE=$(DIR)/passwd
   ```

   to

   ```
   PWFILE=$(DIR)/passwd.nis
   ```

8. Rebuild and propagate the new passwd maps.

   ```
   # /var/yp/ypmake passwd
   ```


© 2003 Hewlett-Packard Development Company, L.P.

11–14. LAB: Configuring NIS

Directions

In this lab exercise, you will work with a team of two to four classmates to configure and test NIS servers and clients in your own NIS domain. Working with the teammates assigned by your instructor, decide on a name for your NIS domain.

Domain Name: _____

Within your domain, you should configure one master server, a slave server, and one or more clients. Decide among yourselves which machine will be your master server, which will be the slave, and which will be the client(s):

Master server: _____

Slave server: _____

Client(s): _____

Note that the examples referenced in the instructions that follow refer to a domain called "california" containing three hosts. Within this sample domain, "sanfran" is the master server, "oakland" is the slave server, and "la" is a client.

Part 1: Configuring an NIS Master Server

The following steps should only be performed on the NIS master server. Do not start configuring the slave or clients until the master configuration is complete.

1. Ensure that your ASCII source files (**/etc/passwd**, **/etc/group**, etc.) are up-to-date. Although the ASCII files may be changed after configuring NIS, it is much easier to make changes now. For the sake of this lab exercise, you may assume that your ASCII source files are already up-to-date.

2. The script used to configure the NIS master server must know ahead of time the name of the domain. Do this by setting your server's NIS domain name with the **domainname** command:

```
# domainname california   # set your domain name
# domainname              # check your domain name
```


© 2003 Hewlett-Packard Development Company, L.P.

3. Next, run the **`ypinit -m`** command to build all the maps for your domain. When asked if you wish to "quit on non-fatal errors," answer "n." **`ypinit`** prompts for a list of slave servers for the domain, then builds all the necessary maps.

   ```
   # ypinit -m
   ```

4. Next, modify the NIS startup configuration file, **`/etc/rc.config.d/namesvrs`**. Enable your machine as an NIS_MASTER_SERVER and define your NIS_DOMAIN. To ensure consistency across the domain, the master should also be configured as a client. Enable NIS_CLIENT functionality as well.

5. Reboot to start NIS on the master.

6. When your machine comes back up again, check to see which processes are running. What NIS-related processes would you expect to see on an NIS master server?

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring an NIS Slave Server

Every NIS domain should have at least one NIS slave server to provide service to the clients if the master becomes unavailable. In subnetted networks, each subnet usually has a separate NIS slave server.

Do not begin this portion of the lab until the master server is fully configured.

1. Start by setting your domain name as you did on the master.

2. Run the **`ypinit -s`** ***`masterserver`*** command, where ***`masterserver`*** is the host name of your master server. This downloads the NIS maps from the master. When asked if you wish to quit on non-fatal errors, answer "**`n.`**"

   ```
   # ypinit -s sanfran
   ```

3. Watch the **`ypinit`** messages. What does the **`ypinit`** do to configure the slave server? (Note: disregard the ethers, bootparams, and netmasks errors generated by **`ypinit`**. These maps are not used in HP-UX, but the **`ypinit`** utility still attempts to download them.)

4. **`ypinit`** should have copied the NIS maps from the master server, and stored them under the slave server's **`/var/yp directory`**. Do an **`ls`** of **`/var/yp`**, and find the subdirectory for your domain. What do you see in your domain's **`/var/yp`** subdirectory?

© 2003 Hewlett-Packard Development Company, L.P.

5. Next, modify the NIS startup configuration file, **`/etc/rc.config.d/namesvrs`**. Enable your machine as an NIS_SLAVE_SERVER and NIS_CLIENT and define your DOMAINNAME.

6. Remove all of your users' entries from your local password file, since NIS will now be providing central administration of your user account information. However, be sure to leave all accounts with userids below 100 in **`/etc/passwd`**. Why might it be important to leave these userids (especially root.) in place?

7. Reboot.

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring NIS Clients

Do not continue on to this step until at least one of your NIS servers has finished booting. Now configure the remaining hosts in your team as NIS clients.

1. Enable NIS client functionality and define your domain name in the **`/etc/rc.config.d/namesvrs`** config file.

2. As you did with your slave server, remove all user entries from **`/etc/passwd`**.

3. Reboot.

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Using NIS Maps

After the system finishes booting, try a few tests to see if your NIS configuration was successful. Since all of your machines in the domain are clients, even the master and slave can try these exercises.

1. The **ypwhich** command tells you which server you are bound to. Which server are you currently bound to?

2. The **ypcat** command displays the contents of NIS maps. Adding the **-k** option also shows the key value associated with each entry in the map files. View the contents your **hosts** map by typing:

   ```
   client# ypcat -k hosts.byname
   client# ypcat -k hosts.byaddr
   client# ypcat hosts
   ```

   "**hosts**" is just an abbreviation for **hosts.byaddr**. To list the other nicknames recognized by **ypcat**, try:

   ```
   client# ypcat -x
   ```

3. You can check the value associated with any key in an NIS map by using the **ypmatch** command:

   ```
   client# ypmatch user1 passwd.byname
   client# ypmatch 0  passwd.byuid
   ```


© 2003 Hewlett-Packard Development Company, L.P.

4. Do the standard UNIX utilities use the NIS? To find out, try logging in as user1. Note that user1 no longer exists in the slave or clients' local password files. Why does this login succeed?

5. Try another system utility. Use **`nslookup`** to determine which IP address is associated with your neighbor's host name. Does **`nslookup`** appear to use NIS? How can you tell?

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Updating NIS Maps

1. Start with an easy NIS update. Log in as user1 on the client and type **passwd** to change user1's password.

2. Is the password change reflected in the password map on the master, the slave, or both? Use the **yppoll** command to check the order number on the master and the slave servers:

```
# yppoll -h slave passwd.byname
# yppoll -h master passwd.byname
# yppoll -h slave passwd.byuid
# yppoll -h master passwd.byuid
```

   Are the order numbers the same?

3. Try another change on the client. Create a user account in the **/etc/passwd** file on the client, then **ypcat** the passwd map again. Does **ypcat** show the new account? Explain.

```
client# useradd donald
client# ypcat passwd
```

4. What happens if you make your changes to **/etc/passwd** on the master server instead of the client? Try it. Add user **donald** to the master server's **passwd** file. Then **ypcat** the **passwd** map and explain the result.

```
master# useradd donald
master# ypcat passwd
```


© 2003 Hewlett-Packard Development Company, L.P.

5. On the master, do whatever is necessary to rebuild the **passwd** map and propagate the updates to the slave server. Use **ypcat** to ensure this worked properly.

6. What happens if an NIS slave is down when the master attempts to push an update? Try it and find out.

   - Stop CDE.
   - Shutdown the LAN card on the slave.
   - Add user pluto to the master's **/etc/passwd** file.
   - **ypmake** the passwd map on the master. (Be patient.)

   Did **ypmake** warn you that the slave was down?

7. Bring the slave's LAN card back up again, then do whatever is necessary on the slave to update the maps. Note: **ypxfr** does not recognize the NIS nicknames.

8. Is any harm done if you **ypxfr** a map that is already up-to-date? Try it. From the slave, try another **ypxfr** on **passwd**. What happens? Why might this behavior be advantageous?

© 2003 Hewlett-Packard Development Company, L.P.

Part 6: Securing Clients and Slave Servers with Password/Group Escape Entries

Currently, anyone listed in the NIS passwd map can log onto your NIS client. Your goal in this exercise is to modify your client configuration so only user1-user3 are allowed to log in (as well as **`root`**, of course).

1. Start out by adding the escape entries to the client's **`/etc/passwd`** file that would allow user1-user3, but no other NIS map users, to successfully log in.

2. Did your escape entry have the desired effect? Can your client **`su`** to user1's account? Can your client **`su`** to user6's account? Why can user6 still log in?

3. Create a new **`/etc/nsswitch.conf`** file for yourself with the entries required to recognize escape characters in **`/etc/passwd`** and **`/etc/group`**.

4. Try logging in with the user1 and user6 usernames again. What happens now?

© 2003 Hewlett-Packard Development Company, L.P.

Part 7: (Optional) Securing the NIS Master Server

The escape entries you used in the previous part of the exercise provide a convenient mechanism for restricting access to NIS clients and slaves. However, some special NIS configuration changes are required if you wish to restrict access to the master server.

1. Why can't you restrict access to the master server by simply deleting all the user lines from **`/etc/passwd`**, so only the root and basic system userids remain?

2. Follow the steps suggested in the notes to restrict access to the master server so only root can log in.

3. Try logging into your master server as user3. This should fail.

© 2003 Hewlett-Packard Development Company, L.P.

Part 8: (Optional) When Things Go Wrong . . .

1. During the remainder of the lab, you will be asked to shutdown your LAN card several times. Execute the following command to shutdown CDE before proceeding:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. What happens if the NIS master server is unreachable for a period? Take down the LAN card on your master server.

3. Can clients still access the maps? From the client, **`ypcat passwd`** and explain the result. (Be patient.)

4. Can changes be made to the maps while the server is down? Log in as user1 on the client and try changing the password with **`passwd`**. What happens? (Be patient.)

5. Now take down the slave's LAN card, too.

© 2003 Hewlett-Packard Development Company, L.P.

6. Try a **`ypcat`** on **`passwd`**. What happens? (Be patient. Once you see a few error messages, press return to get back to a prompt.)

7. Bring the LAN cards on both servers back up again.

© 2003 Hewlett-Packard Development Company, L.P.

Part 9: (Optional) Troubleshooting NIS

You have seen what happens when a client is no longer able to communicate with the NIS servers. What can you do to troubleshoot the problem?

1. What NIS-related process(es) must be running on the client? Do a **`ps -ef`** to ensure that the necessary processes are actually running.

2. See if your client can still access the NIS maps. Try a **`ypcat passwd`** and see what happens (be patient).

   When an NIS server goes down, the client's first access may eventually time out and generate an error. However, **`ypbind`** immediately attempts to bind to another NIS server on the subnet. Try another **`ypcat passwd`** and see what happens. Did the **`ypcat`** succeed this time?

3. There are a number of RPC daemons that must be running on an NIS server in order for clients to be able to access the NIS maps. How can the client see if these RPCs are registered and available?

Part 10: Cleanup

Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Module 12 — Configuring DNS

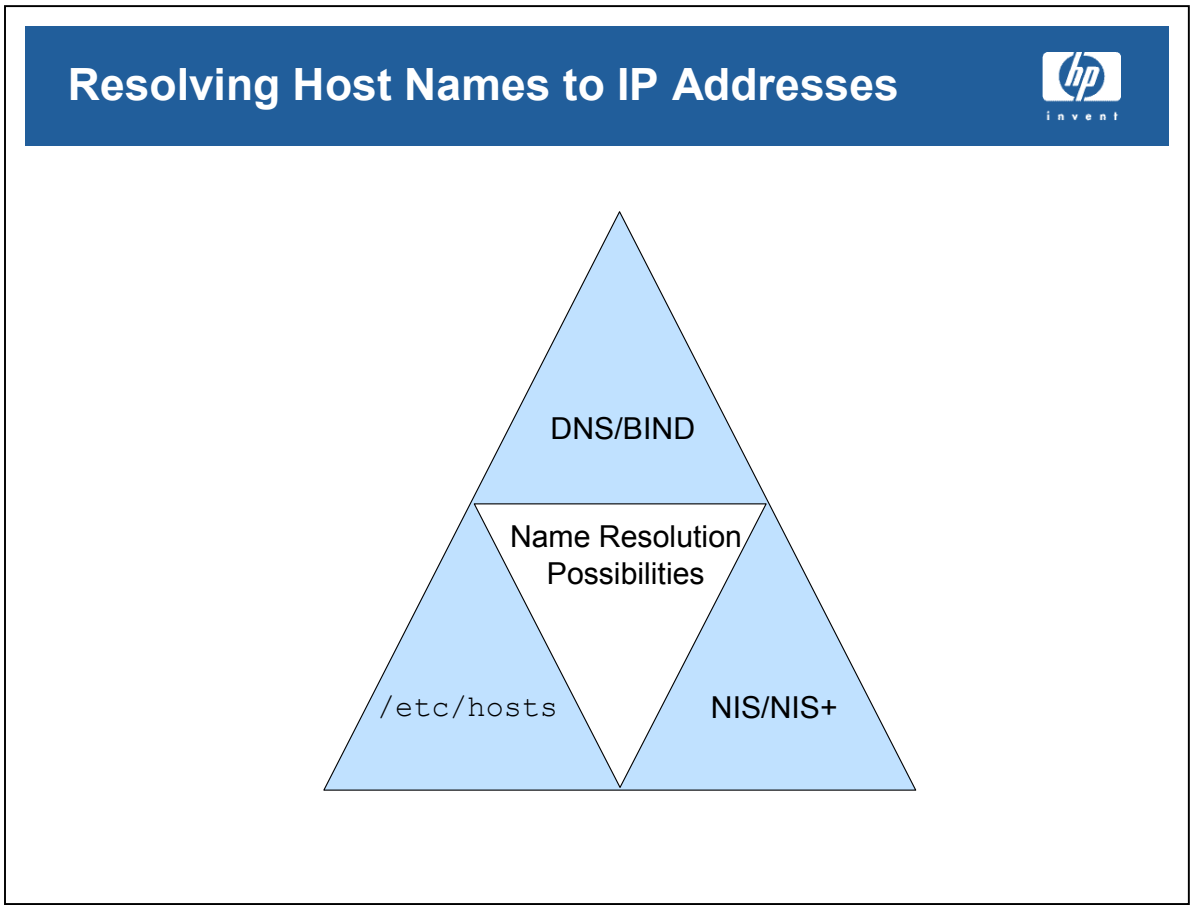
Objectives

Upon completion of this module, you will be able to do the following:

- Compare and contrast the three approaches to host name resolution:
  - **`/etc/hosts`**
  - NIS
  - DNS/BIND
- Configure a primary DNS server using the **`hosts_to_named`** command.
- Configure a secondary name server.
- Configure a cache-only name server.
- Configure a resolver-only host.
- Configure the **`/etc/nsswitch.conf`** file.
- Add or remove a host in the DNS database, using the **`hosts_to_named`** command.
- Troubleshoot DNS using **`nslookup`** and **`nsquery`**.
- Describe the purpose and format of the following configuration files:
  - **`/etc/rc.config.d/namesvrs`**
  - **`/etc/named.conf`**
  - **`/etc/resolv.conf`**

© 2003 Hewlett-Packard Development Company, L.P.

12–1. SLIDE: Resolving Host Names to IP Addresses



Student Notes

Every packet that is sent across an IP network must contain a destination IP address. However, users often prefer to identify destination hosts by hostname rather than IP address, because IP addresses are difficult to remember. Most applications allow users to enter destinations as hostnames, then automatically translate those hostnames to IP addresses using the **`gethostbyname()`** resolver library function.

Many applications use a related function called **`gethostbyaddr()`** to translate IP addresses back into hostnames. For instance, when the NFS mount daemon receives a mount request from a client, the daemon must determine which client initiated the request. **`rpc.mountd`** checks the source IP address included in the incoming packet, converts it to a hostname via the **`gethostbyaddr()`** function and then verifies that the resulting hostname is included in the export list for the requested file system.

The resolver routines may use several different mechanisms to resolve hostnames and IP addresses. Each method is described briefly below.

`/etc/hosts`

© 2003 Hewlett-Packard Development Company, L.P.

When the Internet was small, hostname resolution was handled exclusively via the **/etc/hosts** file. Each entry in the **/etc/hosts** file has an IP address followed by the hostname associated with that IP address. As networks grew larger and more geographically disbursed, it became increasingly difficult to maintain consistent, updated **hosts** files across all systems on the Internet. A more scalable solution was needed!

NIS

The Network Information Service simplifies host file maintenance by requiring all hosts on a subnet to query a central NIS server for hostname lookups. Thus, using NIS, the administrator needs only to manage one central **hosts** map instead of hundreds of **/etc/hosts** files on individual hosts. Unfortunately, NIS does not scale well. The NIS hosts map becomes increasingly unwieldy when it grows beyond a few hundred hostnames.

DNS/BIND

As the number of hosts on the Internet grew into the tens of thousands, a more flexible, more scalable solution was required. The Domain Name Service (DNS) makes it possible to manage millions of hostnames and IP addresses efficiently, and has become the primary name resolution mechanism used on the Internet today.

There have been several implementations of DNS over the years. UNIX systems typically use the Berkeley Internet Name Domain (BIND) implementation that was developed at UC Berkeley. Microsoft systems use a different DNS implementation. Fortunately, both DNS implementations use the same protocols for exchanging DNS information.

BIND has gone through many revisions over the years. Since many of these updates include patches to security vulnerabilities, it is important to update BIND as new versions become available. The BIND version number is included in the header information at the top of the **/usr/sbin/named** executable. Use the **what** command to extract this version information:

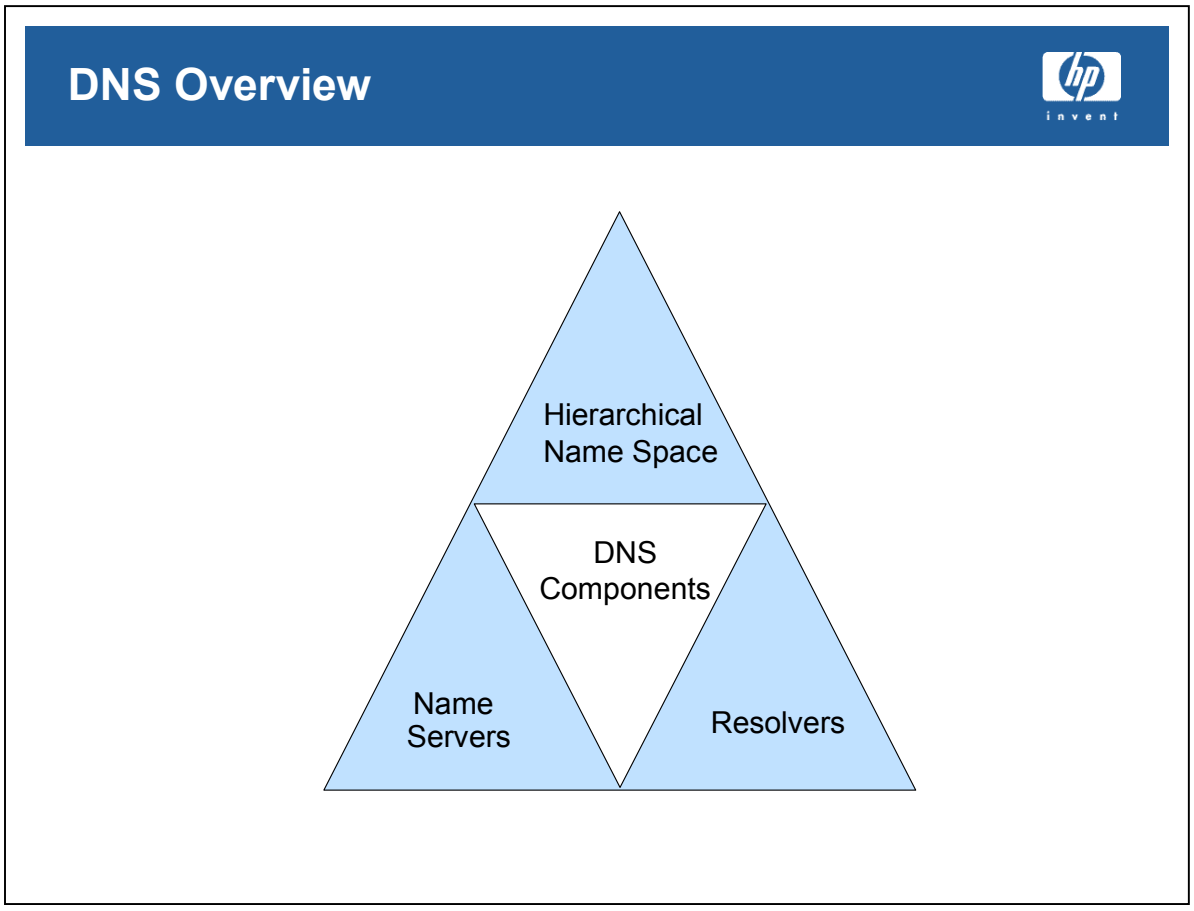
```
# what /usr/sbin/named
```

The latest HP-supported version of BIND is usually available on the **http://software.hp.com** website. The HP-supported version of BIND usually lags slightly behind the most current version of BIND. You can download and compile the latest version of BIND source code yourself from **http://www.isc.org/**.

The examples in this workbook were taken from a system running BIND 8.1.2.

© 2003 Hewlett-Packard Development Company, L.P.

12–2. SLIDE: DNS Overview



Student Notes

There are several important components in the DNS/BIND architecture:

- DNS uses a "Hierarchical Name Space" to group related hosts together into DNS "domains" in much the same way that UNIX uses a hierarchical file system structure to group related files together into directories. Using a hierarchical name space makes it possible to delegate responsibility for portions of the name space to other entities. For instance, Hewlett Packard has been delegated responsibility for all hostnames ending in **`hp.com`**.

- DNS name servers are specially configured hosts on the Internet that are able to resolve hostnames to IP addresses for other client hosts. There are thousands of DNS name servers on the Internet today, each of which is responsible for a small portion of the overall DNS name space.

- Hosts on the Internet use DNS "Resolver Libraries" to send hostname and IP lookup queries to DNS name servers. Any time a user uses **`telnet`**, **`ftp`**, or another network service to access other hosts by hostname, the application uses the **`gethostbyname()`** and **`gethostbyaddr()`** resolver library routines to send a query to a hostname resolution service. The HP-UX resolver routines are able to do lookups using the

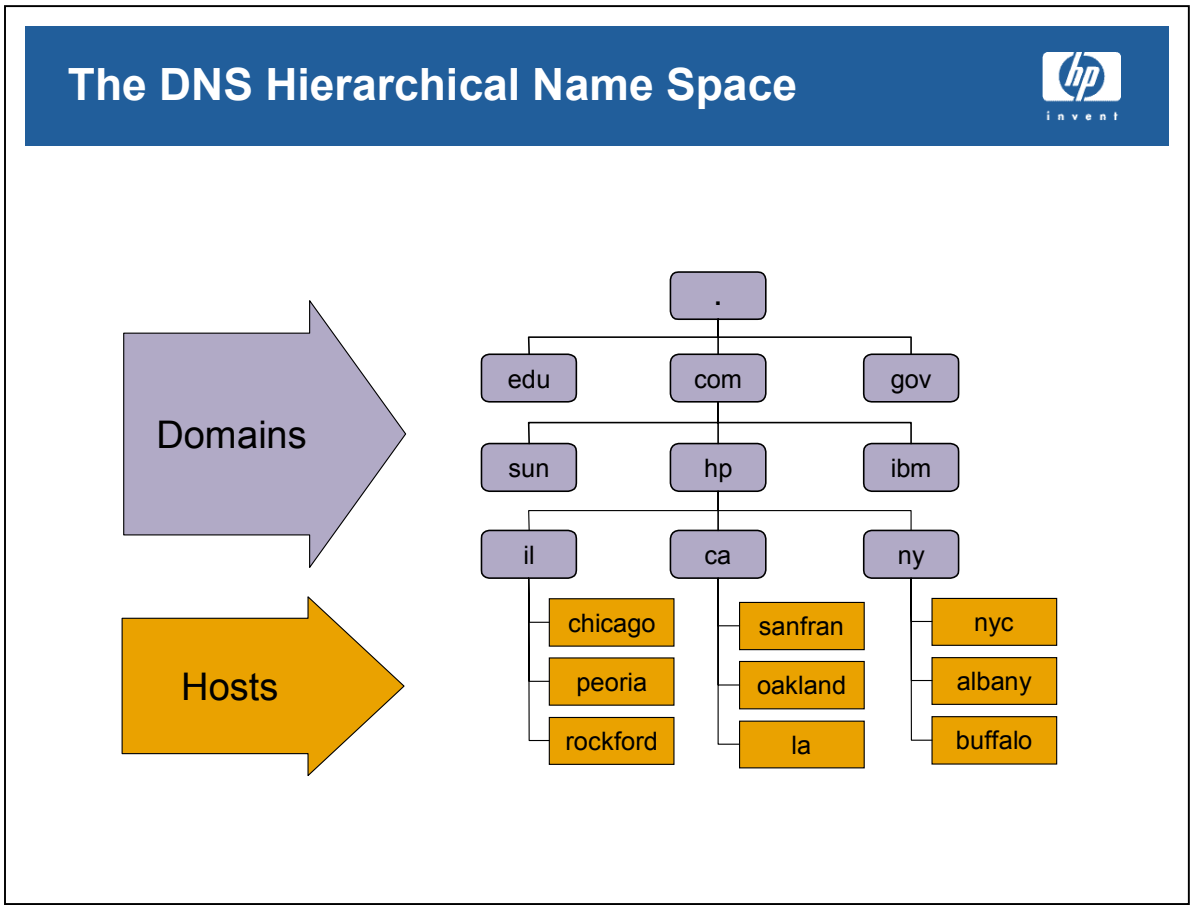
© 2003 Hewlett-Packard Development Company, L.P.

`/etc/hosts` file, NIS, or DNS. You can choose which lookup service or services you want your resolver to use for hostname resolution.

Each of these components will be discussed in detail later in the module.

© 2003 Hewlett-Packard Development Company, L.P.

12–3. SLIDE: The DNS Hierarchical Name Space



Student Notes

The traditional **`/etc/hosts`** file name resolution mechanism used a "flat" name space; all hostnames were defined in a single monolithic **`/etc/hosts`** file that had to be updated anytime a hostname anywhere on the Internet changed.

DNS was designed to be a distributed name resolution service. Responsibility for resolving hostnames is delegated among thousands of DNS name servers on the Internet. Each of these name servers is granted authority over a small portion of the hostnames in the overall name space. This distributed approach greatly simplifies hostname allocation and management.

The DNS hierarchical name space makes it possible to distribute responsibility for the name space among thousands of name servers by forming logical groupings of hosts called DNS domains. By checking a host's domain name, it is possible to determine which name server is responsible for resolving that host's hostname to an associated IP address. For instance, the name servers for the **`hp.com`** domain are responsible for resolving all hostnames ending in **`hp.com`**. The name servers for the **`ibm.com`** domain are responsible for resolving all host names ending in **`ibm.com`**.

All hosts on the Internet ultimately belong to the root (.) level domain at the top of the hierarchy. The root domain is subdivided into several hundred somewhat smaller domains.

© 2003 Hewlett-Packard Development Company, L.P.

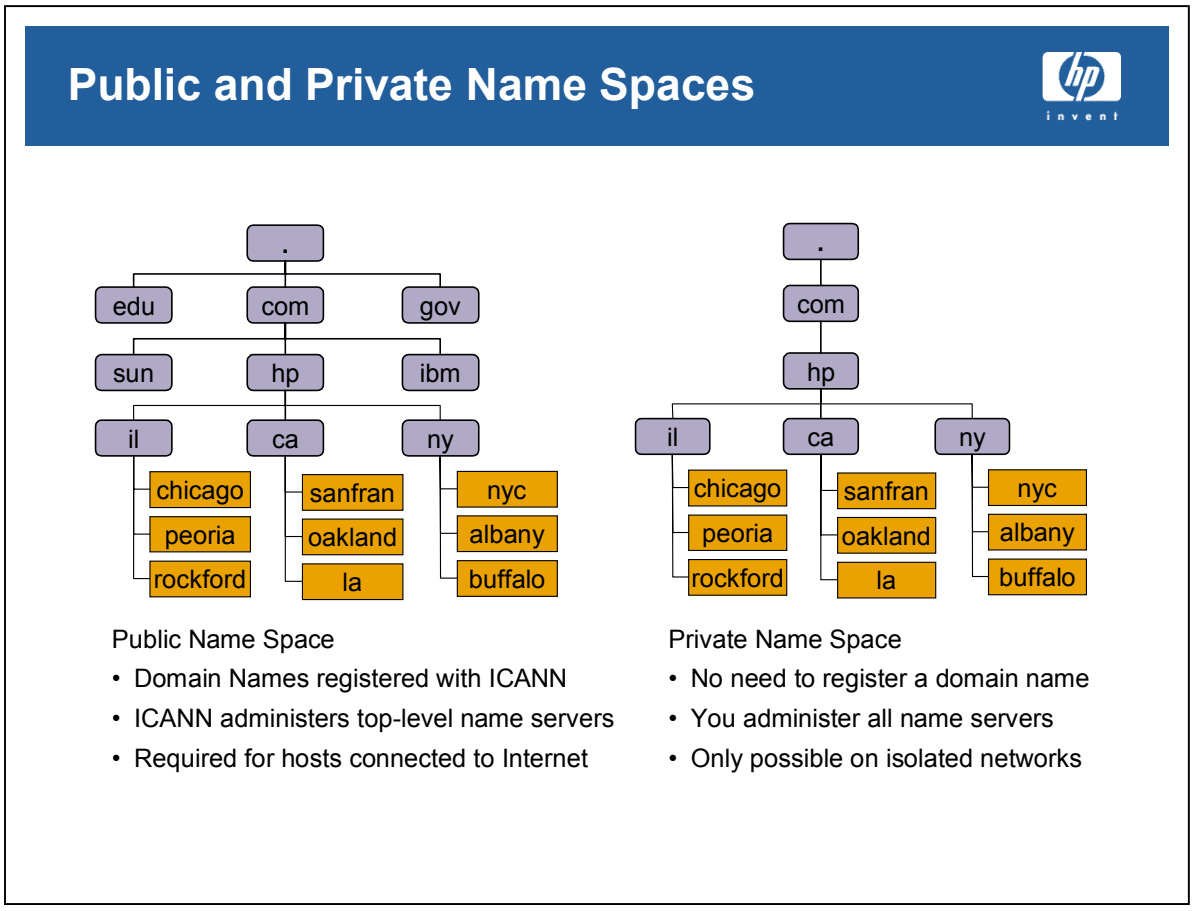
Perhaps the best known of these "top-level" domains are **com** (for commercial entities), **gov** (for U.S. government entities), **edu** (for educational institutions), and **org** (for non-commercial organizations).

Each of these top-level domains is further subdivided into smaller domains. **hp.com**, for instance, is a member of the **com** domain. Many of these domains are subdivided still further. The example on the slide lists several theoretical regional subdomains under hp.com: **ca.hp.com** (for California HP hosts), **il.hp.com** (for Illinois hosts), and **ny.hp.com** (for New York HP hosts). Each organization may choose to subdivide their DNS domain somewhat differently.

Hostnames in the overall DNS name space may be written in one of several different ways. Oftentimes, we identify hosts via their relative, or unqualified, hostnames (for example, **sanfran**, **oakland**, or **la**). In order to unambiguously identify a host on the Internet, though, you should get in the habit of using absolute, or "Fully Qualified Domain Names" (FQDNs) that specify a hostname and the DNS domain that the host belongs to (for example, **sanfran.ca.hp.com.**). Officially, FQDNs always end with a dot representing the root level domain.

© 2003 Hewlett-Packard Development Company, L.P.

12–4. SLIDE: Public and Private Name Spaces



Student Notes

There are two different types of DNS domains. The type of network to which your host is connected will determine how you go about obtaining a domain name for your organization.

The Public Name Space

If your host has a direct connection to the Internet, your host will be part of the DNS Public Name Space. In this case, you must officially register a unique domain name for your organization through one of the accredited domain registrars that is licensed by the Internet Corporation for Assigned Names and Numbers (ICANN). To search the list of accredited registrars, follow the Accredited Registrar link on the **`http://www.icann.org`** web page, or simply ask your ISP to obtain a DNS domain name for you.

When you register your domain, you will be required to provide the IP addresses of one or more DNS name servers that will be authoritative for your domain. When other hosts on the Internet wish to contact hosts in your domain, their hostname resolution requests will be forwarded to one of your authoritative name servers.

After your domain is registered, you can assign hostnames and create subdomains within your domain as you wish. Since you are the delegate authority for your domain, changes

© 2003 Hewlett-Packard Development Company, L.P.

within your domain should be recorded on your authoritative name servers, but need not be recorded with ICANN.

If your organization already has a registered DNS domain name, you should contact your IT department to request a delegated subdomain or hostname.

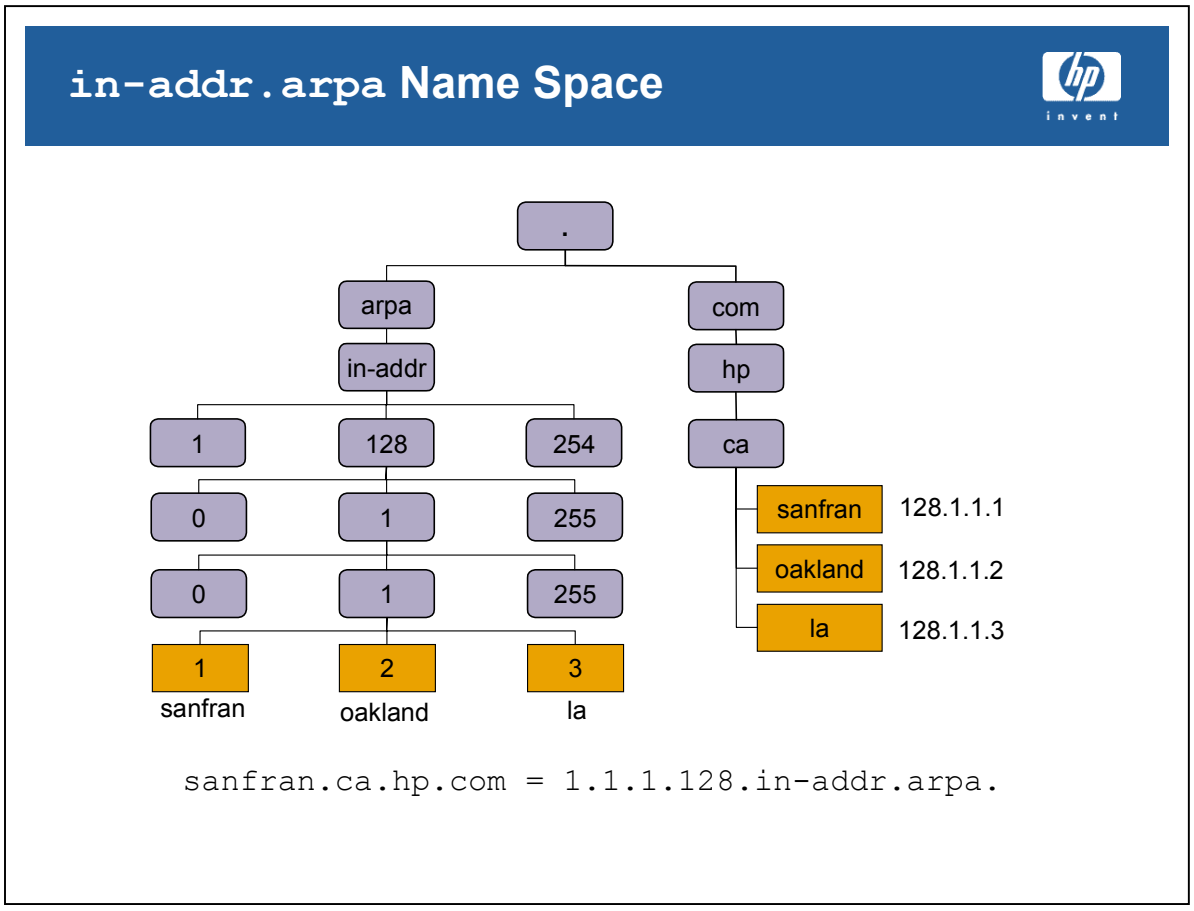
Private Name Spaces

If you manage an isolated network that is not connected to the public Internet, then you may choose to configure a "private" name space. On a private network, you can freely assign hostnames and subdomains however you wish. To facilitate future connections to the public Internet, it is better to apply for an official domain name and follow the DNS naming conventions, even if you do not intend to join the public name space immediately.

In the private name space example on the slide, the private "**.**" domain has only one subdomain: **com**. The private **com** subdomain has only one subdomain: **hp**. The administrator responsible for this network would have to configure a name server for both of these private, upper-level domains, as well as the **hp.com** domain and its delegated subdomains. A single name server could be configured to manage all three of these domains.

© 2003 Hewlett-Packard Development Company, L.P.

12–5. SLIDE: `in-addr.arpa` Name Space



Student Notes

The primary purpose of the DNS name space is to map host names to IP addresses. However, there are situations where applications may request a reverse lookup; given an IP address, a name server may be asked to find the associated hostname. The **`in-addr.arpa`** portion of the DNS name space makes this reverse resolution possible.

Every IP address may be represented as a leaf in the **`in-addr.arpa`** DNS domain. To convert an IP address to its **`in-addr.arpa`** equivalent, simply reverse the order of the IP octets, and append the **`in-addr.arpa`** domain name. The table below shows several examples:

`128.1.1.1 = 1.1.1.128.in-addr.arpa. (sanfran)`
`128.1.1.2 = 2.1.1.128.in-addr.arpa. (oakland)`
`128.1.1.3 = 3.1.1.128.in-addr.arpa. (la)`

Each DNS name server is responsible for a small portion of the **`in-addr.arpa`** domain. If, for instance, all hosts in the **`ca.hp.com`** domain had IP addresses on the **`128.1.1`** subnet,

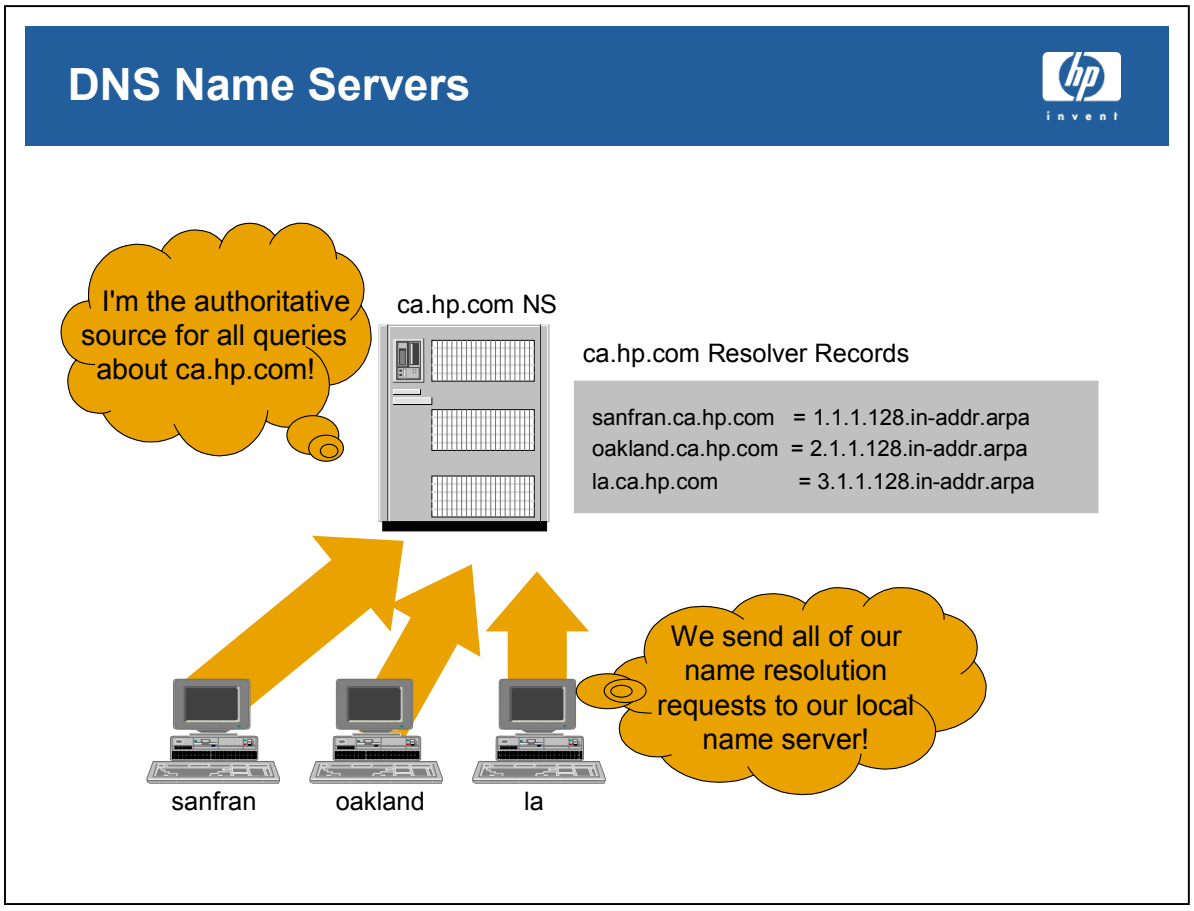
© 2003 Hewlett-Packard Development Company, L.P.

then the **ca.hp.com** name server would also be responsible for the **1.1.128.in-addr.arpa** portion of the **in-addr.arpa** domain.

Name servers for domains that span multiple subnets may be responsible for multiple subdomains under **in-addr.arpa**.

© 2003 Hewlett-Packard Development Company, L.P.

12–6. SLIDE: DNS Name Servers



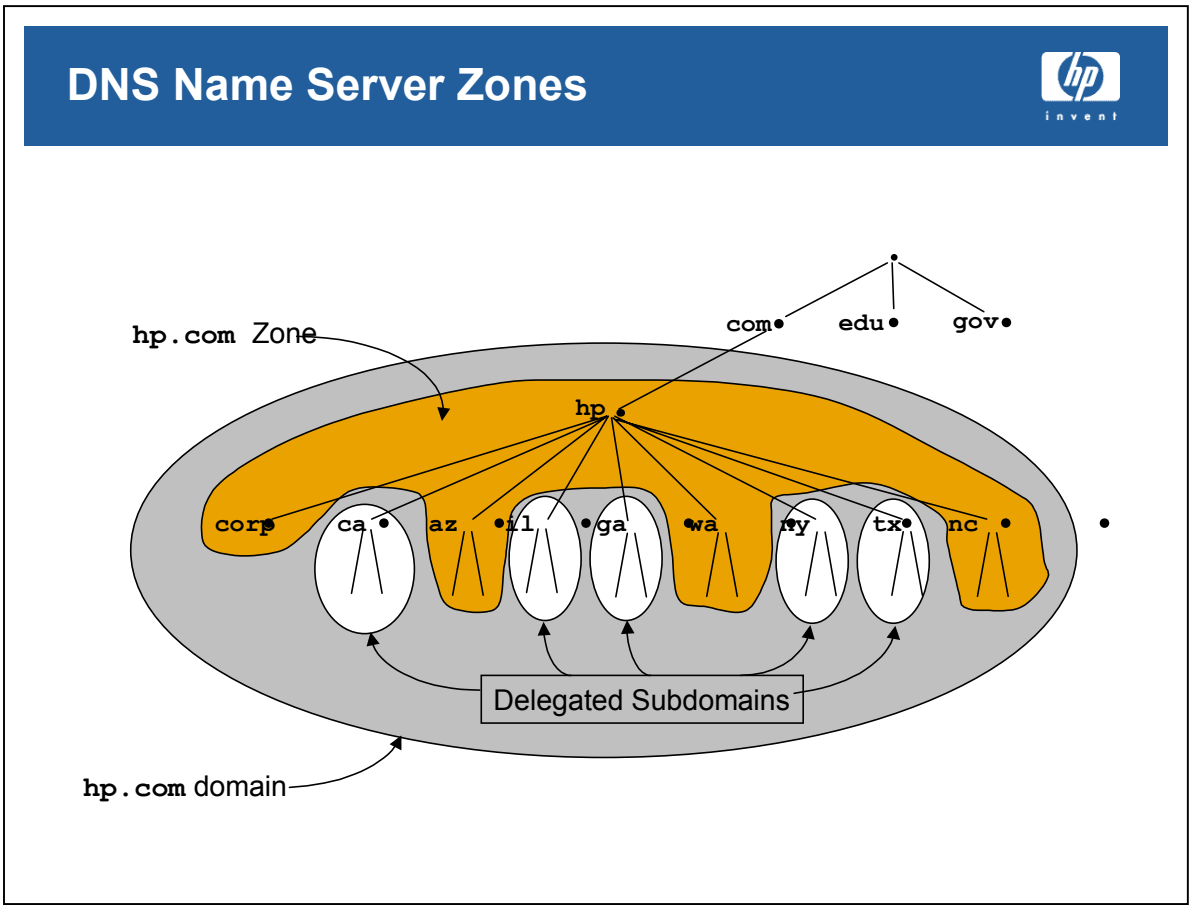
Student Notes

Hosts on the Internet, which have the ability to resolve DNS hostnames to IP addresses and IP addresses to hostnames, are called DNS "Name Servers." DNS clients send their hostname and IP lookup requests to DNS name servers. In some cases, the name server may already know the hostname or IP address that a client has requested in its DNS Resolver Record database. In other cases, however, a name server may need to query other name servers to find the information it needs to answer a client's query.

The BIND implementation of DNS uses a daemon called **`named`** to provide name service for DNS clients.

© 2003 Hewlett-Packard Development Company, L.P.

12–7. SLIDE: DNS Name Server Zones



Student Notes

Every DNS name server maintains a database of DNS "Resolver Records" that fully describes a portion of the DNS name space. The portion of the name space for which a name server has a full set of resolver records is known as the server's "Zone."

In some cases, a name server's zone may include all of the hosts in a single domain. For instance, if the **hp.com** domain had a single name server, then all hosts in the **hp.com** domain would also be included in the **hp.com** zone of authority.

Oftentimes, though, a name server may delegate responsibility for a portion of its domain to other name servers. In the example on the slide, the **ca.hp.com** is a delegated subdomain with its own DNS name server. Since the **hp.com** name server has delegated responsibility for California to another name server, the **ca.hp.com** subdomain is excluded from the **hp.com** name server's zone of authority. **il.hp.com**, **ga.hp.com**, **ny.hp.com**, and **tx.hp.com** are similarly excluded from the **hp.com** name server's zone of authority.

az.hp.com, **wa.hp.com**, and **nc.hp.com** are non-delegated subdomains that do not have their own name servers. Instead, the **hp.com** name server includes these subdomains in its zone of authority.

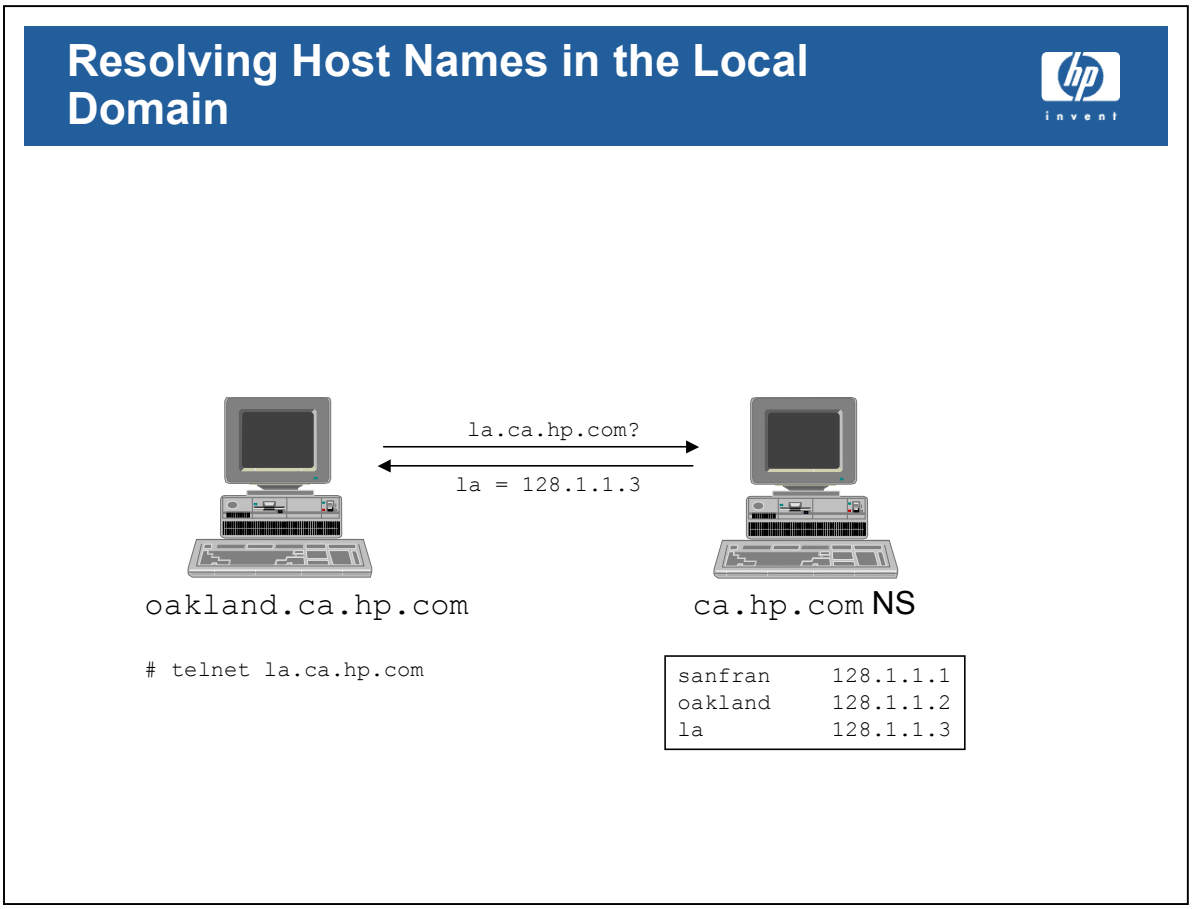
© 2003 Hewlett-Packard Development Company, L.P.

In summary, each name server is able provide the following authoritative information:

- The name server's own hostname and IP address
- The hostnames and IP addresses of all hosts within the name server's zone of authority
- The IP addresses of the name server's delegated subdomain name servers

© 2003 Hewlett-Packard Development Company, L.P.

12–8. SLIDE: Resolving Host Names in the Local Domain



Student Notes

Each time you invoke an application and specify a target host by name, the application uses the **`gethostbyname()`** system to resolve that hostname to an IP address. The resolver must perform several tasks for the application:

- First, the resolver must determine if the local node is using DNS, NIS, or **`/etc/hosts`**. Our example here will assume that DNS is the client's preferred name resolution mechanism. The **`/etc/nsswitch.conf`** file determines which lookup source the client uses. It will be discussed later in this chapter.

- If DNS is the preferred hostname resolution mechanism, and the user provided an unqualified hostname, the resolver builds a search list of possible fully qualified hostnames that the user may be attempting to resolve. For instance, if the user types "**`telnet la`**," the resolver routine must guess which domain host **`la`** might be in. The resolver builds a list of possible fully qualified hostnames using the domain search list specified in the client's **`/etc/resolv.conf`** file.

© 2003 Hewlett-Packard Development Company, L.P.

If the client's search list included **ca.hp.com**, **il.hp.com**, and **hp.com**, the resulting list of possible fully qualified hostnames might look something like this:

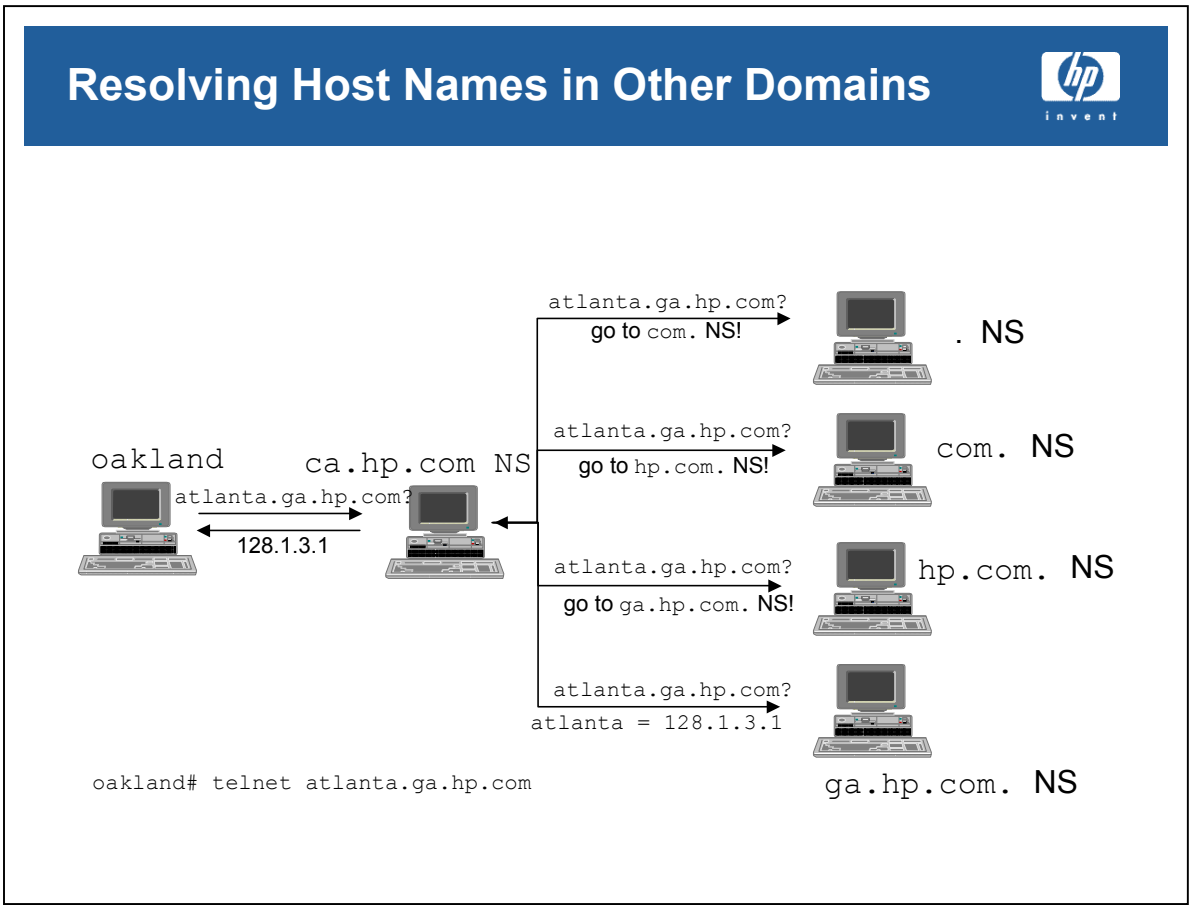
```
la.ca.hp.com
la.il.hp.com
la.hp.com
```

If the user provides a fully qualified host name (with a dot "**.**" at the end), the resolver routine simply attempts to resolve that hostname without consulting the domain search list. **/etc/resolv.conf** is described in detail later in this chapter.

- Finally, the resolver queries the local name server to determine if any of the hostnames generated in the previous step can be successfully resolved into an IP address. The **/etc/resolv.conf** file may specify up to three name servers. If the first name server fails to respond within 75 seconds, the resolver tries the second name server, and eventually the third. If DNS is unconfigured, or if the name servers fail to respond, the resolver may automatically resort to using NIS or the local **/etc/hosts** file, depending on the "switch" mechanism defined in **/etc/nsswitch.conf**. This switch mechanism is described in detail later in this chapter.

© 2003 Hewlett-Packard Development Company, L.P.

12–9. SLIDE: Resolving Host Names in Other Domains



Student Notes

When accessing hostnames in other domains, the DNS client still sends the lookup request to the local DNS name server. If a name server receives a query regarding a hostname that is not included in the name server's own local zone data, the name server automatically performs a recursive search for the hostname in other domains.

The sequence of events that occur when performing the recursive search are described below:

1. The root server is queried. It provides the best answer it can: the address of the name server closest to the destination.

2. The local DNS server then queries the name server suggested by the root-level server, which responds with a referral to another server. After following several such referrals, the local name server will eventually reach the name server whose zone of authority includes the requested hostname. The answer provided by this server is said to be an **authoritative answer**. The local DNS name server caches the addresses of all the name servers, as well as the final answer.

© 2003 Hewlett-Packard Development Company, L.P.

3. If another client queries the local name server regarding the same hostname, the local DNS server responds immediately with the cached data. Since this cached information may be outdated, this is said to be a "non-authoritative" answer. Servers flush their cached records on a regular (configurable) basis.

Notice a DNS name server initially knows only the hostnames and IP addresses of the hosts within its own zone of authority, and the IP addresses of the root level name servers. A name server does not initially know the addresses of its sibling name servers in other portions of the domain. However, as the name server's cache builds over time, the name server will be able to answer more and more queries non-recursively using information stored in cache.

Example on Slide

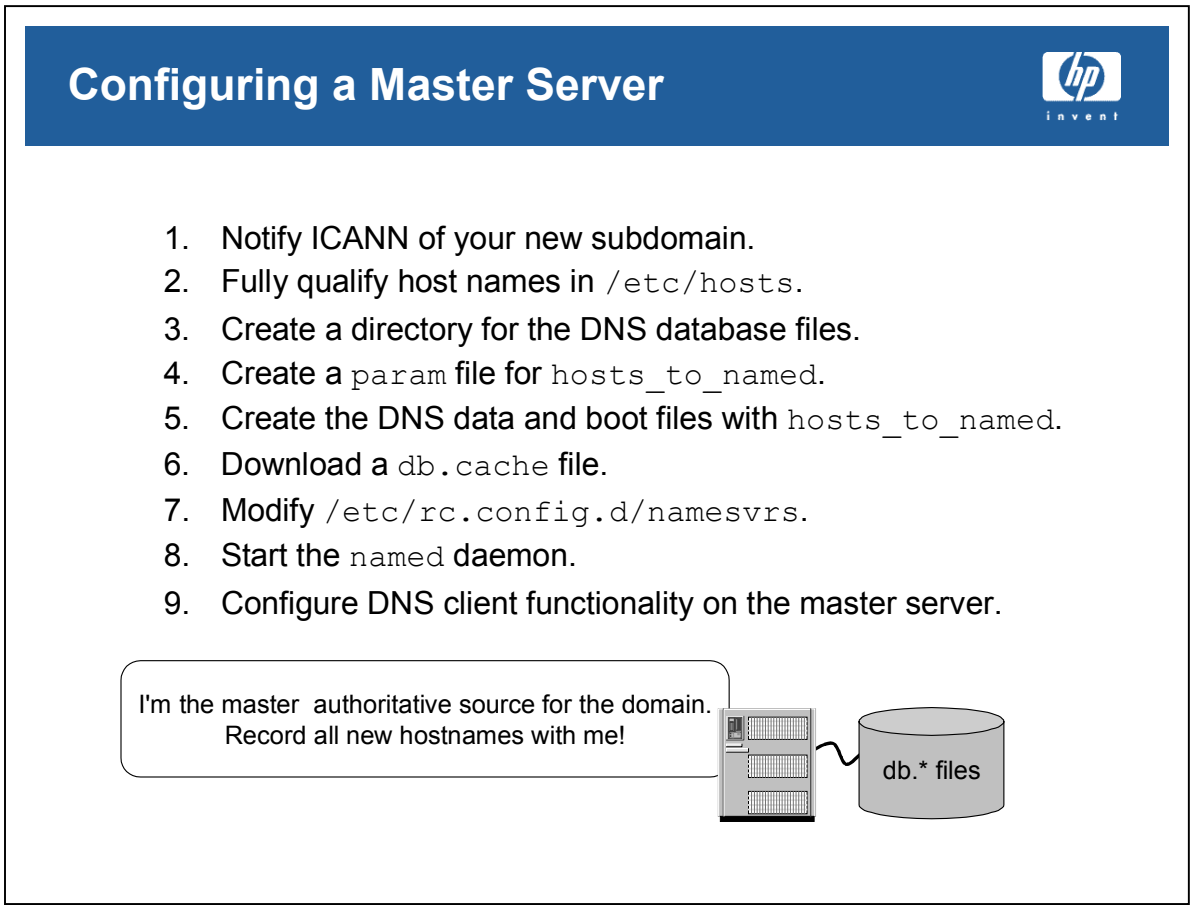
In the example on the slide, client **`oakland`** requests **`atlanta.ga.hp.com`**'s IP address from the **`ca.hp.com`** name server.

Since the local DNS name server for the **`ca.hp.com`** domain does not know **`atlanta`**'s IP address, it queries the root level name server (**`.`**). The root name server suggests a query to the **`com`** name server, which suggests a query to the **`hp.com`** name server, which suggests a query to the **`ga.hp.com`** name server. Finally, **`ga.hp.com`** responds with an authoritative answer, which the **`ca.hp.com`** name server relays back to **`oakland`**.

In the meantime, the **`ca.hp.com`** name server caches all of this information for future queries.

© 2003 Hewlett-Packard Development Company, L.P.

12–10. SLIDE: Configuring a Master Server



Student Notes

Every DNS zone must have one "Master Server" (also known as the "Primary Name Server"). The master server is the authoritative source for information about hosts in the zone. Any hostnames that are added to the domain must be added to the master server's zone database files, and any hosts that are removed from the domain must be removed from the master's zone database files. The master server can also delegate responsibility for subdomains to other name servers.

Configuring the Master Server

The step-by-step procedure for configuring a master server is shown below. The notes assume that **`sanfran`** is being configured as a master server for the domain **`ca.hp.com`**.

1. Register your domain name. In order for others on the Internet to resolve the names of hosts in your domain, you must officially register your domain name. Go to the **`http://www.icann.org`** website for a list of officially accredited domain registrars. If you are creating a subdomain in a domain already established by your company, you may have to deal with your internal IT department instead. In either case, you will probably need to provide a contact name for your subdomain, your subdomain name, and the name and address of your master server.

© 2003 Hewlett-Packard Development Company, L.P.

2. Fully qualify host names in **/etc/hosts**. The **hosts_to_named** utility provided with HP-UX can create the DNS data files on your master server using the information already in your **/etc/hosts** file. In order for this to work though, all of the entries in your **hosts** file need to be converted to fully qualified host names. The old host names can be used as aliases. If you wish, you can delete lines in the **/etc/hosts** file that refer to domains for which your name server is not responsible. (Note, however, that the **localhost** entry must remain.) The example below shows the changes that would be required on **sanfran**:

```
# vi /etc/hosts
127.0.0.1 localhost
128.1.1.1 sanfran.ca.hp.com sanfran
128.1.1.2 oakland.ca.hp.com oakland
128.1.1.3 la.ca.hp.com      la
```

3. Create a directory for the DNS database files. The **hosts_to_named** program will create several DNS data files. These data files are typically stored in a directory called **/etc/named.data**. Create the **/etc/named.data** directory manually with **mkdir**.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
# cd /etc/named.data
```

4. Create a **param** file for **hosts_to_named**.

   The **hosts_to_named** utility is a powerful tool for building DNS database. **hosts_to_named** looks for a **param** file to determine which domains your name server will serve.

   - Include a **-d** entry for each domain for which this name server will be responsible. Since some name servers serve multiple domains, you may have multiple **-d** entries.
   - Include a **-n** entry for each (sub)net included in this domain. Since many domains include hosts on several subnets, you may have multiple **-n** entries.
   - The **-b** option determines where your DNS boot configuration file will be stored. **/etc/named.conf** is the standard location.
   - The next slide will discuss "Secondary Servers," which serve as backups for the master server. The secondary (or slave) servers will need to download a configuration file containing the IP address of the master server and other information about the domain. The **-z** option in the **param** file creates this configuration file for the slave servers.
   - Other options may be specified in this file as well. See the **hosts_to_named** man page for details.

   The **param** file for the **sanfran** name server looks like this:

```
# vi param
    -d ca.hp.com        # Use your domain name(s) here
    -n 128.1.1          # Use your subnet address(es) here
    -z 128.1.1.1        # Use your master server's IP here
    -b /etc/named.conf
```


© 2003 Hewlett-Packard Development Company, L.P.

5. Create the DNS data and boot files with **hosts_to_named.** The **hosts_to_named** utility automatically creates all the DNS data files needed to resolve host names and IP addresses in your domain using your **/etc/hosts** file, and the options defined in your **param** file.

```
# hosts_to_named -f param
Translating /etc/hosts to lower case ...
Collecting network data ...
   128.1
Creating list of multi-homed hosts ...
Creating "A" data (name to address mapping) for net 128.1.1 ...
Creating "PTR" data (address to name mapping) for net 128.1.1 ...
Creating "MX" (mail exchanger) data ...
Building default named.boot file ...
Building default db.cache file ...

WARNING: db.cache must be filled in with
         the name(s) and address(es) of the
         rootserver(s)

Building default boot.sec.save for secondary servers ...
Building default boot.sec for secondary servers ...
Building default boot.cacheonly for caching only servers ...
done
```

6. The **hosts_to_named** utility creates all of the necessary DNS database files except one. You must manually populate the **db.cache** file with the addresses of the root-level name servers. You can **ftp** a file containing the current root server list from **ftp://ftp.rs.internic.net/domain/root.zone**. Since the list of root servers changes from time to time, you will need to download updates on a monthly basis.

7. For the exercises that we do in class, we will download this file from the instructor station, rather than the internic.

```
# ftp 128.1.0.1
> get /etc/named.data/db.cache
> quit
```

8. Modify **/etc/rc.config.d/namesvrs**. In order to ensure that the name server daemon, **named**, starts during the boot process, set the NAMED variable in the **/etc/rc.config.d/namesvrs** configuration file to "1".

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

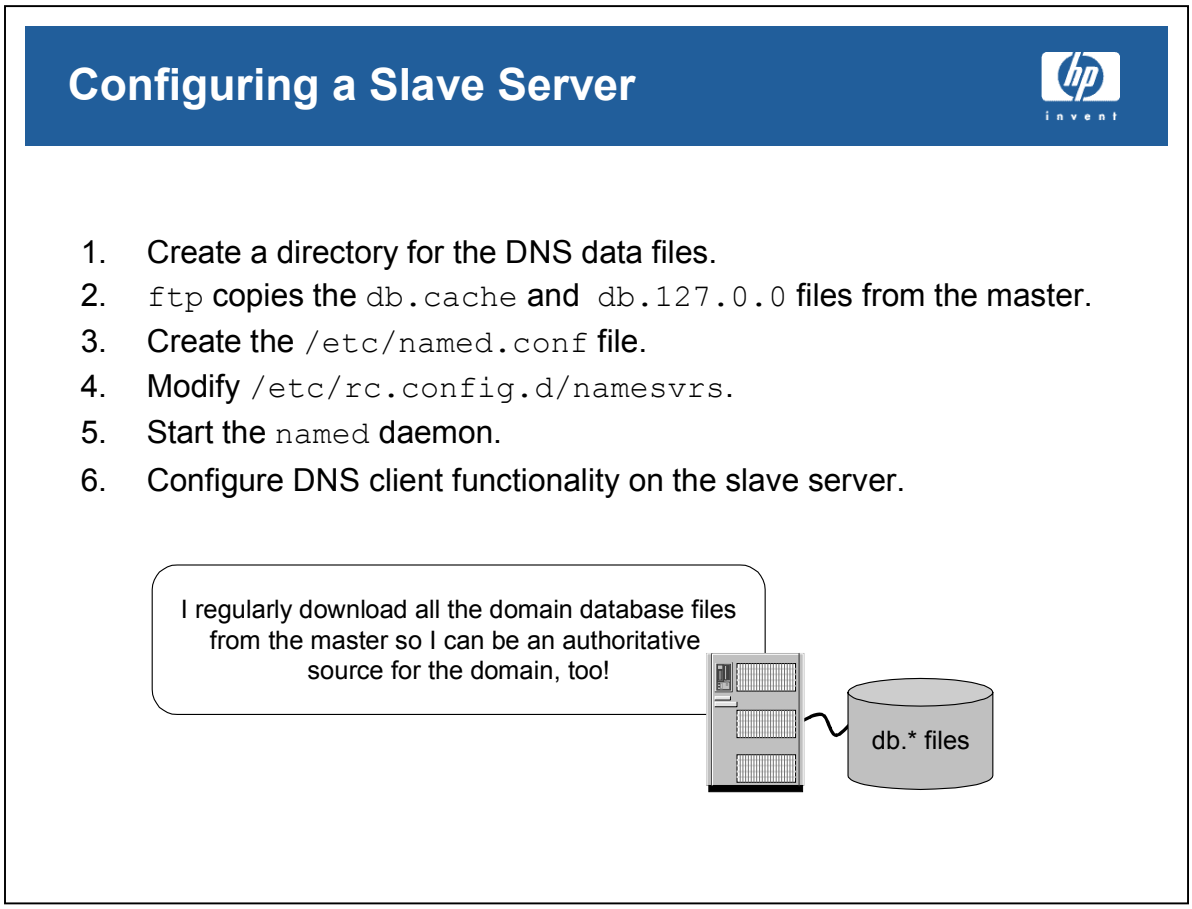
9. Start the **named** daemon. A reboot is not required.

```
# /sbin/init.d/named start
```

10. Configure DNS client functionality on the master server. Most DNS servers are also DNS clients. DNS client configuration is covered later in this chapter.

© 2003 Hewlett-Packard Development Company, L.P.

12–11. SLIDE: Configuring a Slave Server



Student Notes

Most domains have one or more slave servers (also called "secondary" name servers) in addition to the domain master server. At boot time and at regular intervals thereafter, the slave servers do a "zone transfer" to download copies of the zone database files from the master server. Some slave servers store the zone data in data files on disk, while other simply retain the downloaded data in memory.

Slave servers serve two purposes. First, slave servers provide a backup name server source if the master server becomes unavailable. Second, slave servers reduce the load on the master by handling some queries from clients' resolvers.

Configuring a Slave Server

To create a slave server perform the following steps: (the steps below would be used to configure a slave server for the **`ca.hp.com`** domain, if **`sanfran.ca.hp.com`** is the master server for the domain)

1. On the slave server, create a separate directory for the database and configuration files. Most slave servers store local copies of the domain's DNS database files in the **`/etc/named.data`** directory.

© 2003 Hewlett-Packard Development Company, L.P.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
```

2. **ftp** copies the **db.cache** and **db.127.0.0** from the master. The slave server will copy the remaining **db.*** files (if needed) over, when the named daemon is first initialized and spawned.

```
# ftp 128.1.1.1
> get /etc/named.data/db.cache
> get /etc/named.data/db.127.0.0
```

There are two different types of slave servers. Some slave servers store copies of the master's database files on disk. Other slave servers simply copy the master's database information directly into cache at boot time. The first approach allows the slave server to answer clients' queries even if the master server is unreachable when the slave server boots. The second approach saves some disk space.

3. Create the **/etc/named.conf** file. The **named** daemon determines where its DNS database files are stored by consulting the **/etc/named.conf** file at startup. Running **hosts_to_named** on the master server automatically creates a boot file for the slave servers. **ftp** the boot file from the master server, then move it to its proper location on the slave. You can download an appropriate file from the master server.

```
# ftp 128.1.1.1       # Use your master server's IP here
> get /etc/named.data/conf.sec.save
> quit
# mv /etc/named.data/conf.sec.save /etc/named.conf
```

If you do not want to maintain disk-based copies of the DNS database files on your slave server, then download and install the **/etc/named.data/conf.sec** file instead.

4. Modify **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

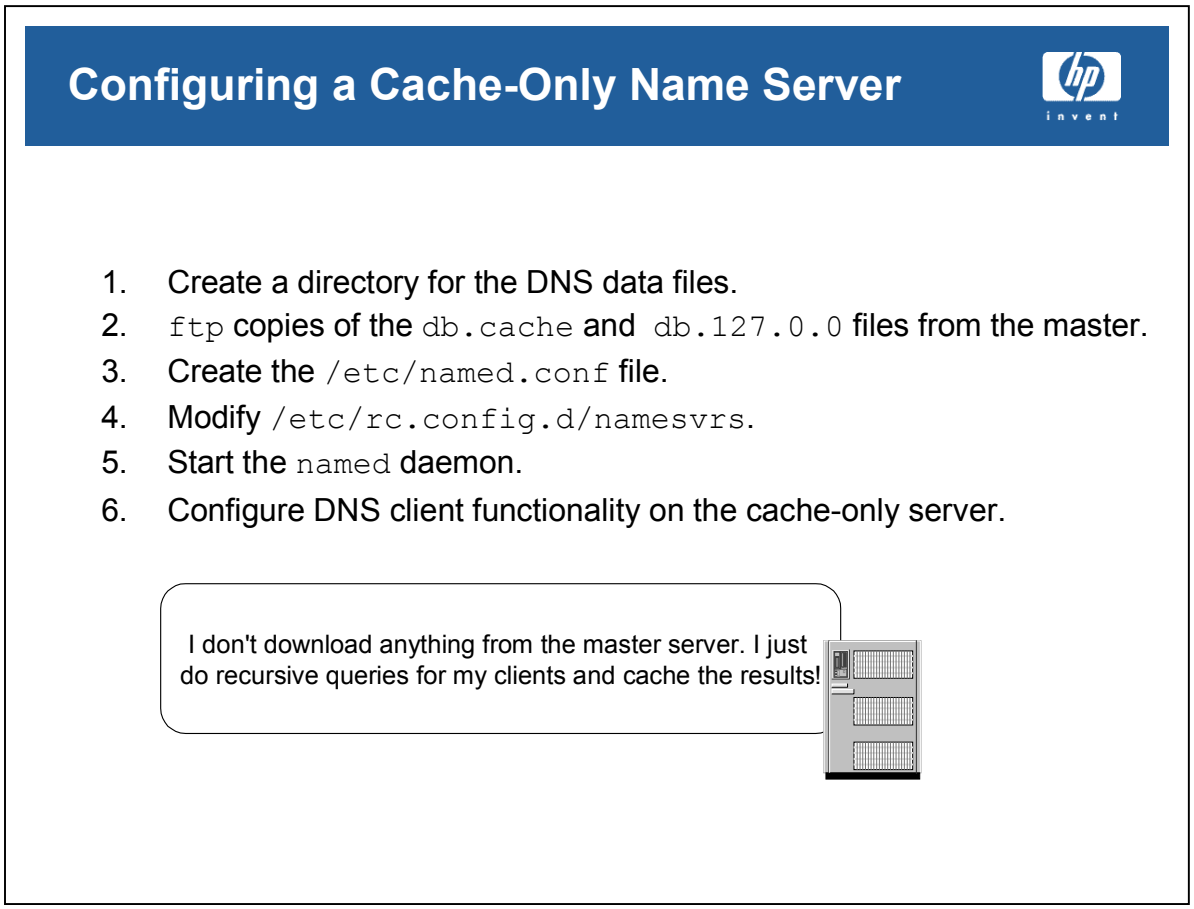
5. Start the **named** daemon. A reboot is not necessary.

```
# /sbin/init.d/named start
```

6. Configure DNS client functionality on the slave server. Most DNS servers are also DNS clients. DNS client configuration is covered later in this chapter.

© 2003 Hewlett-Packard Development Company, L.P.

12–12. SLIDE: Configuring a Cache-Only Name Server



Student Notes

Master and slave servers both maintain authoritative database records for one or more domains. A cache-only name server does not maintain authoritative information for *any* domains (except 127.0.0.1). Any time a cache-only server receives a query regarding a new hostname, it must do a recursive query to find the desired information. However, every lookup on behalf of a client adds another entry to the server's cache. Over time, as the cache grows, fewer and fewer client requests result in recursive queries.

Some administrators configure a cache-only server on each subnet to minimize network traffic across firewalls and routers, yet without the hassle of managing dozens of full-fledged slave servers.

Configuring a Cache-Only Server

To create a cache-only server, perform the following steps: (the steps below would be used to configure a slave server for the **`ca.hp.com`** domain, if **`sanfran.ca.hp.com`** is the master server for the domain)

1. On the cache-only server, create a separate directory for the database and configuration files. Most slave servers store local copies of the domain's DNS database files in the **`/etc/named.data`** directory.

© 2003 Hewlett-Packard Development Company, L.P.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
```

2. **ftp** copies of the **db.cache** and **db.127.0.0** files from the master. The cache-only server only needs to be able to resolve the **loopback** address and find the root-level name servers. Cache-only servers do not need copies of all of the other **db.*** files

```
# ftp 128.1.1.1                # Use your master server's IP here
> get /etc/named.data/db.cache
> get /etc/named.data/db.127.0.0
> quit
```

3. Create the **/etc/named.conf** file. The **named** daemon determines where its DNS database files are stored by consulting the **/etc/named.conf** file at startup. Running **hosts_to_named** on the master server automatically creates a boot file for the slave servers. **ftp** the boot file from the master server, then move it to its proper location on the slave. You can download an appropriate file from the master server.

```
# ftp 128.1.1.1      # Use your master server's IP here
> get /etc/named.data/conf.cacheonly
> quit
# mv /etc/named.data/conf.cacheonly /etc/named.conf
```

4. Modify **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

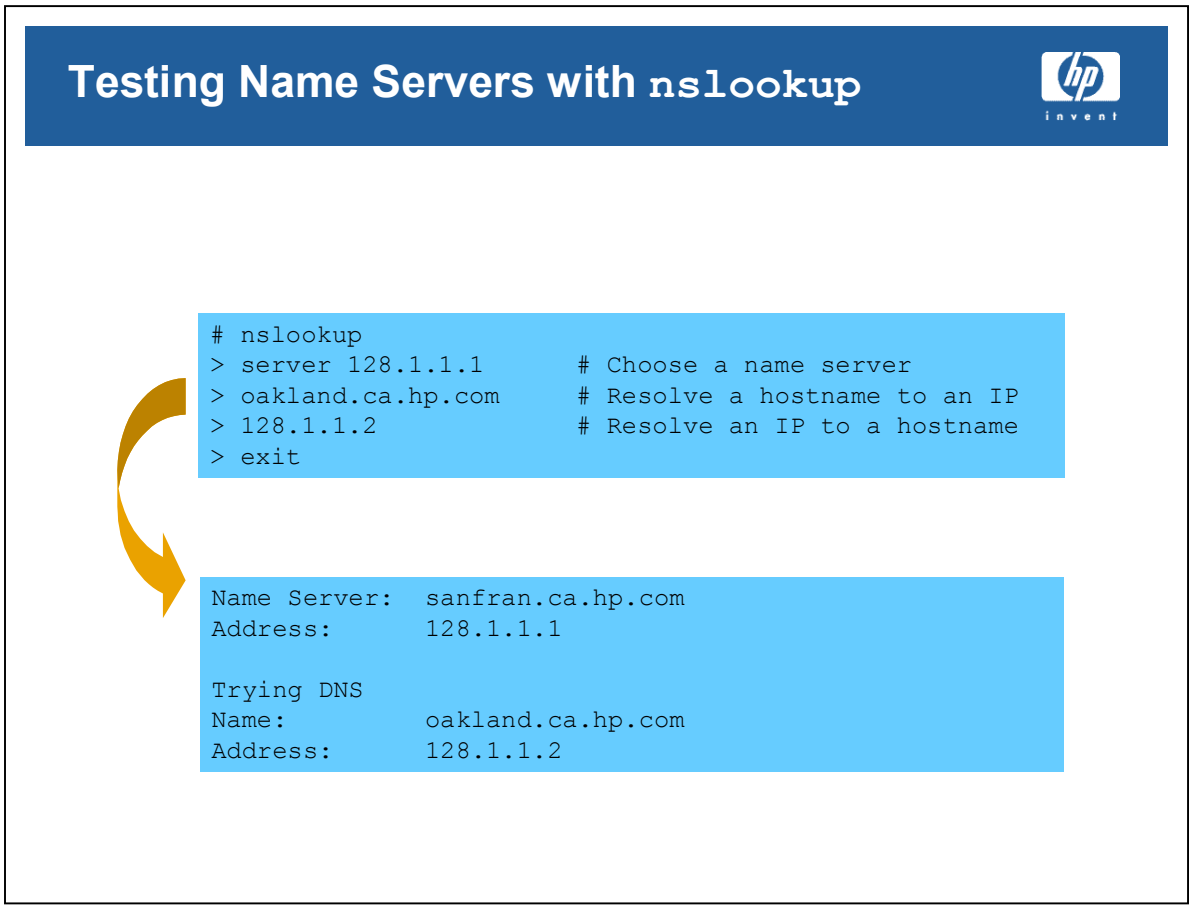
5. Start the **named** daemon. A reboot is not necessary.

```
# /sbin/init.d/named start
```

6. Configure DNS client functionality on the cache-only server. Most DNS servers are also DNS clients. DNS client configuration is covered later in this chapter.

© 2003 Hewlett-Packard Development Company, L.P.

12–13. SLIDE: Testing Name Servers with `nslookup`



Student Notes

You can ensure that your DNS name servers are functioning properly using the **`nslookup`** command. If your host has already been configured with DNS client functionality, simply type:

```
# nslookup corp.hp.com          (simple host name lookup)
# nslookup 128.1.0.1            (simple IP address lookup)
```

`nslookup` uses your default name server as configured in **`/etc/resolv.conf`** and responds back with the IP or host name you requested.

Alternately, if you haven't yet configured DNS client functionality, or if you wish to override the default name server listed in **`/etc/resolv.conf`**, you may wish to run **`nslookup`** interactively:

```
# nslookup
> server 128.1.1.1    (try some lookups using the master server)
> corp.hp.com
> 128.1.0.1
> server 128.1.1.2    (now test the slave server, too)
> corp.hp.com
```


© 2003 Hewlett-Packard Development Company, L.P.

```
> 128.1.0.1
> exit
```

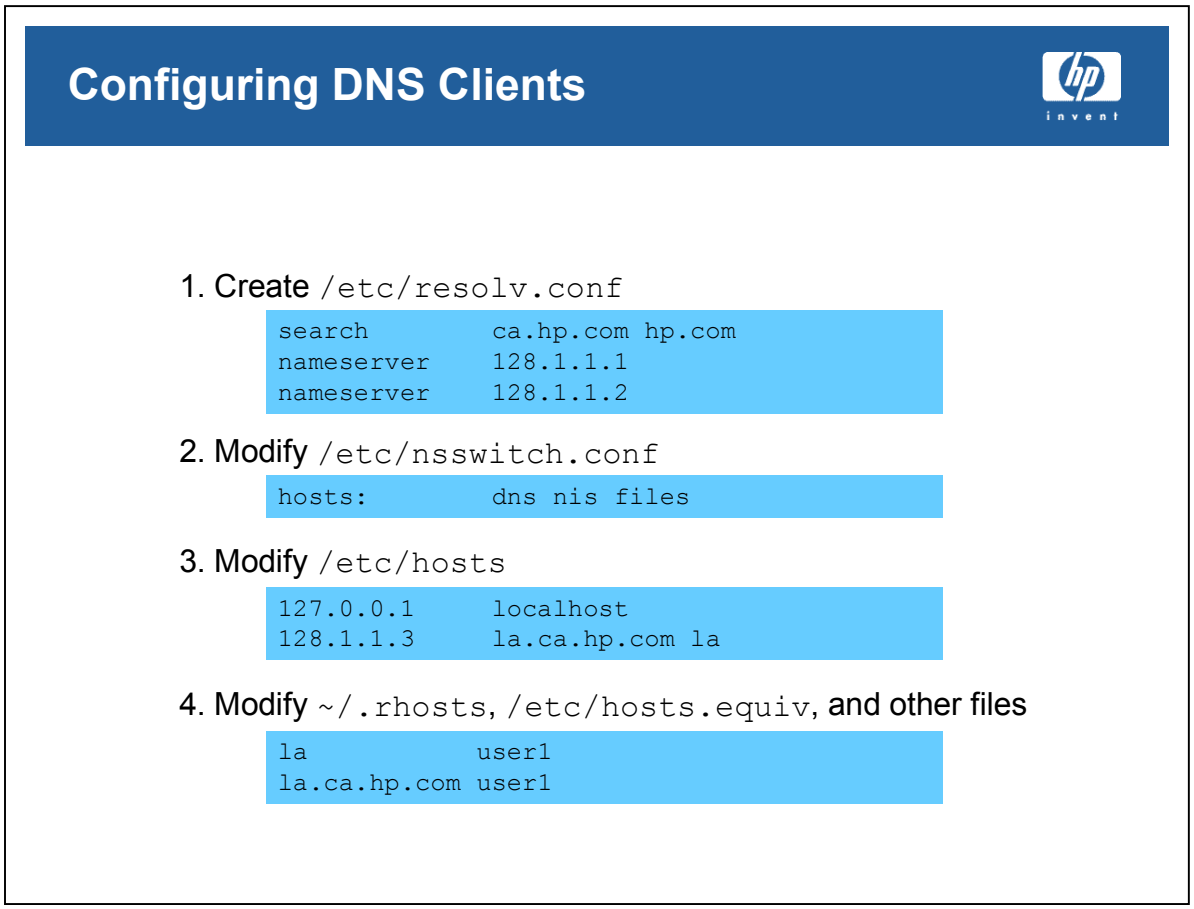
There are many other commands available within **nslookup** for troubleshooting your DNS name servers. At the "**>**" prompt, you can enter a "**?**" for a list of available tools within **nslookup**.

Question

You may notice that **nslookup** sometimes returns a "Non-authoritative answer." In fact, if you look up the same host name twice, you may notice that only the second response from **nslookup** is marked as "Non-authoritative." Can you guess why?

© 2003 Hewlett-Packard Development Company, L.P.

12–14. SLIDE: Configuring DNS Clients



Student Notes

All hosts within a DNS domain, including the master and slave servers, should be configured as DNS clients. Configuring a host as a DNS client ensures that the host's resolver routines resolve host names and IPs using a designated DNS name server rather than the local **`hosts`** file. The steps required to configure a host as a DNS client are described below.

1. Modify the resolver configuration file. The configuration file for the system host name resolver routines is called **`/etc/resolv.conf`**. The **`resolv.conf`** file has two important components:

   a. Creating a **`resolv.conf`** "search" list

   The **`search`** keyword in **`/etc/resolv.conf`** defines a list of domains the resolver should search when resolving host names. At the very least, you should list your own host's domain immediately after the keyword "search." For added flexibility, you should also list other domains.

   Including other domains in the search list saves your users the hassle of fully qualifying host names for machines in the listed domains. For example, since the **`resolv.conf`** file shown below includes **`ca.hp.com`** in the search list, users could telnet to **`sanfran`** by simply typing **`telnet sanfran`**. Accessing **`atlanta`**,

© 2003 Hewlett-Packard Development Company, L.P.

however, would require a fully qualified host name, since **ga.hp.com** is not included in the search list. Include your users' most frequently referenced domains in the search list.

```
# vi /etc/resolv.conf
  search ca.hp.com hp.com # replace ca.hp.com with your domain name
```

b. Adding "name server" entries to **/etc/resolv.conf**

Your local resolver must be told which name server to use when resolving host names and IP addresses. You may configure up to three name server IP addresses in the **/etc/resolv.conf** file; if the first name server listed fails to respond, the resolver will automatically try the second name server.

Since the resolver will always access the DNS servers in the order in which they are listed in **resolv.conf**, you can provide some measure of load balancing by alternating the order in which the servers are listed. On some hosts, list the master server first; on others list the slave server first.

```
# vi /etc/resolv.conf
  search ca.hp.com hp.com # replace ca.hp.com with your domain name
  nameserver 128.1.1.1    # replace 128.1.1.1 with your master's IP
  nameserver 128.1.1.2    # replace 128.1.1.2 with your slave's IP
```

2. Modify **/etc/nsswitch.conf**.

HP-UX can resolve host names using the local hosts file, NIS, or DNS. The **/etc/nsswitch.conf** file determines which source the resolver uses for name resolution. If you do not have an **/etc/nsswitch.conf** file, DNS is the default name resolution source anyway, and you can skip this step. If you have a **hosts** entry in your **/etc/nsswitch.conf** file, ensure that DNS is the first source listed. A later slide in the chapter will discuss **/etc/nsswitch.conf** in more detail.

```
# cat /etc/nsswitch.conf
        ...
        hosts:          dns files
        ...
```

Once **/etc/resolv.conf** and **/etc/nsswitch.conf** have been configured, the resolver immediately begins to use DNS for name resolution.

3. Modify **/etc/hosts.**

Since most host names will now be resolved using the DNS server, you may choose to remove many of the entries in **/etc/hosts**. However, you should retain some critical entries in case the name servers become unavailable. At a minimum, retain the localhost entry, and your own host name.

© 2003 Hewlett-Packard Development Company, L.P.

On the master server, retain all the host entries for your name server's zone. They are required by the **hosts_to_named** utility. Make sure that the host names that remain in **/etc/hosts** are fully qualified. You may also wish to include the "non-qualified" host names as aliases. On **la.ca.hp.com**, the modified hosts file might look like this:

```
# vi /etc/hosts
     127.0.0.1   localhost
     128.1.1.3   la.ca.hp.com  la
```

4. Modify **.rhosts**, **/etc/hosts.equiv**, etc.

Any utilities that do reverse resolution to convert the IPs of incoming packets to host names must be updated with the DNS domain name appended to each host name. If the following files exist, fully qualify each of the host names they contain:

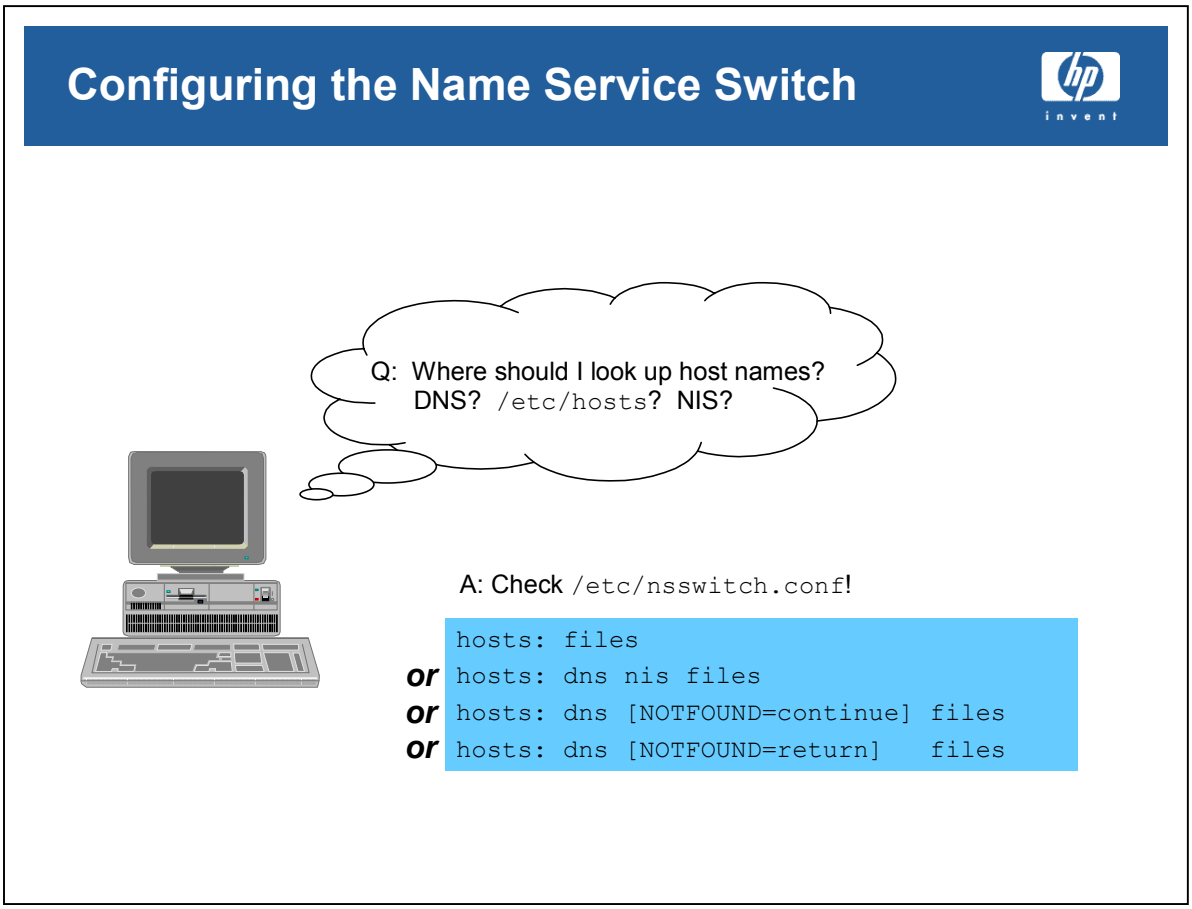
```
~/.netrc
 /etc/hosts.equiv
 /var/adm/inetd.sec
```

For example, **la**'s updated **.rhosts** file might be updated to contain:

```
# vi ~/.rhosts
     oakland.ca.hp.com
     sanfran.ca.hp.com
     la.ca.hp.com
```


© 2003 Hewlett-Packard Development Company, L.P.

12–15. SLIDE: Configuring the Name Service Switch



Student Notes

Applications, utilities, and daemons on an HP-UX box frequently need to resolve IP addresses to host names, UIDs to user names, and GIDs to group names. In fact, these are just a few of the many types of names and addresses that need to be resolved in a UNIX environment.

HP-UX can resolve many of these addresses using a variety of "databases." Host names, for instance, may be resolved to IP addresses via the local **/etc/hosts** file, DNS, or NIS. Somehow, the administrator needs to be able to specify if and when each of these resources should be referenced. This is the purpose of the **/etc/nsswitch.conf** file.

A Simple /etc/nsswitch.conf Entry

Each line in **/etc/nsswitch.conf** begins with a keyword identifying the type of lookup defined by that line. Some common values in this first field include: "hosts," "passwd," and "group." Our discussion here will concentrate on the "hosts" line in **/etc/nsswitch.conf**. The "hosts" line determines how the system should resolve host names to IPs, and IPs to host names.

© 2003 Hewlett-Packard Development Company, L.P.

The remaining fields on the "hosts" line in **/etc/nsswitch.conf** determine which sources should be used when resolving host names and IP addresses. In its simplest form, the hosts line may take one of the following forms:

```
hosts: files          (Consult only the local /etc/hosts file.)
```

or

```
hosts: dns            (Consult only DNS - never consult /etc/hosts!)
```

On real systems, though, things become more complicated. Many administrators prefer to define a "fallback" mechanism. If the DNS server is down, for instance, you may want your machine to try to resolve host names via the local hosts file. **/etc/nsswitch.conf** makes this possible.

Defining a Fallback Mechanism in /etc/nsswitch.conf

If you wish, you can list multiple sources for host name lookups. For instance, you can choose to use the following:

```
hosts: dns files
```

This line says that the host name resolver routines should resolve host names first via DNS. If the DNS nameserver finds the host name requested, the resolver need look no further. If, however, the DNS nameserver is unavailable or does not recognize the requested host name, the resolver automatically falls back on the local **/etc/hosts** file for host name lookups.

If you are also a member of an NIS domain, you may wish to use the following line, which causes the resolver to try all three lookup sources until it finds the host name or IP address it is looking for.

```
hosts: dns nis files
```

Understanding the /etc/nsswitch.conf Fallback Mechanism

You may wish to define more explicitly, what the resolver should do if a lookup via a particular source fails. Sending a query to one of the lookup sources that you have may yield any one of four different results:

SUCCESS	Source found the requested entry.
NOTFOUND	Source responded "no such entry."
UNAVAIL	Source is not configured.
TRYAGAIN	Source is configured, but the server is not responding.

© 2003 Hewlett-Packard Development Company, L.P.

When the resolver receives one of these responses, you can configure it to react in one of two ways:

`continue`	Try the next source in the list.
`return`	Quit searching, do not consult other sources.

By default at version 11.x, if a "hosts" entry exists in **`/etc/nsswitch.conf`**, the resolver will march through all of the sources listed in **`/etc/nsswitch.conf`** until the desired host name is found. In other words, the default behavior looks like this:

```
SUCCESS=return
NOTFOUND=continue
UNAVAIL=continue
TRYAGAIN=continue
```

Consider the following simple example:

```
hosts: dns files
```

This says that the resolver should try DNS first. If DNS recognizes the requested host name, then use the IP address returned by DNS. If DNS is unconfigured, or if the DNS server doesn't respond in a timely manner, or if the DNS server simply doesn't recognize the requested host name, then the resolver should fall back on the local **`/etc/hosts`** file.

More Explicitly Defining the Fallback Mechanism

If you wish, you may explicitly state the action the resolver should take if a source lookup results in a "SUCCESS," "NOTFOUND," "UNAVAIL," or "TRYAGAIN" condition. Consider the following example:

```
hosts: dns [NOTFOUND=return] files
```

With this entry in your **`/etc/nsswitch.conf`** file, the resolver will attempt host name lookups first via DNS. **`NOTFOUND=return`** means that if the DNS name server responds to a query, but doesn't have any record of the host name in question, the resolver will quit rather than fall back on **`/etc/hosts`**. Since the **`nsswitch.conf`** file does not explicitly state what should occur if the DNS lookup results in a SUCCESS, UNAVAIL, or TRYAGAIN, the resolver uses the default actions for these results:

`SUCCESS=return`	*(default)*
`NOTFOUND=return`	*(as defined in* **`/etc/nsswitch.conf`***)*
`UNAVAIL=continue`	*(default)*
`TRYAGAIN=continue`	*(default)*

Thus, the **`UNAVAIL=continue`** and **`TRYAGAIN=continue`** lines ensure that if DNS is unable to respond for one reason or another, the host can still do lookups via the local **`/etc/hosts`** file.

What if `/etc/nsswitch.conf` Does Not Exist at 11.x?

The discussion up to this point has assumed that some sort of "hosts" line exists in your **`/etc/nsswitch.conf`** file. However, you may discover that your system either does not

© 2003 Hewlett-Packard Development Company, L.P.

have an **/etc/nsswitch.conf** file, or has an **/etc/nsswitch.conf** file without a "hosts" line. If there is not a valid "hosts" line in the **nsswitch.conf** file at version 11.x, then the system uses the following host lookup policy:

```
hosts: dns [NOTFOUND=return TRYAGAIN=return] nis [NOTFOUND=return] files
```

In other words, DNS is referenced first. NIS will only be consulted if DNS is unconfigured or unresponsive. The local hosts file, then, will only be consulted if NIS, too, is unconfigured.

The full list of default actions used by HP-UX 11.x when **/etc/nsswitch.conf** does not exist is shown below:

```
passwd:    files nis
group:     files nis
hosts:     dns [NOTFOUND=return TRYAGAIN=return] nis [NOTFOUND=return] files
networks:  nis [NOTFOUND=return] files
protocols: nis [NOTFOUND=return] files
rpc:       nis [NOTFOUND=return] files
publickey: nis [NOTFOUND=return] files
netgroup:  nis [NOTFOUND=return] files
automount: files nis
aliases:   files nis
services:  nis [NOTFOUND=return] files
```

What if /etc/nsswitch.conf Does Not Exist at 10.x?

The notes above describe the switch behavior at HP-UX 11.x. At version 10.x, the defaults for the **/etc/nsswitch.conf** file were somewhat different.

If **/etc/nsswitch.conf** doesn't exist on a 10.x system, the following policies are used:

```
hosts:     dns nis files
services:  nis files
protocols: nis files
networks:  nis files
rpc:       nis files
netgroup:  nis files
```

Furthermore, the default actions were somewhat different, too:

```
SUCCESS=return
NOTFOUND=return
UNAVAIL=continue
TRYAGAIN=return
```

See the 10.x **switch(4)** man page for more information.

© 2003 Hewlett-Packard Development Company, L.P.

Creating a New `/etc/nsswitch.conf` File

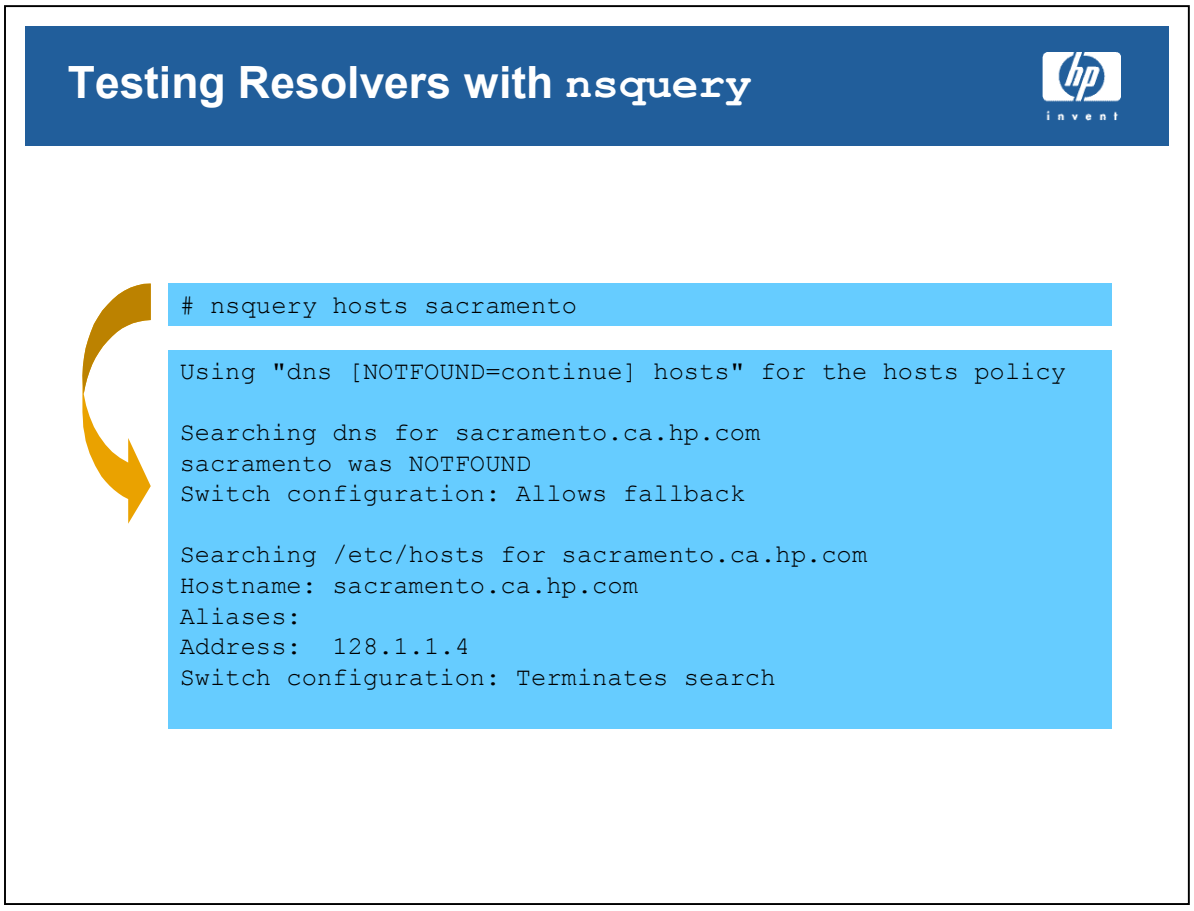
If you don't currently have an **`/etc/nsswitch.conf`** file, you can either create the file yourself using **`vi`**, or copy one of the sample **`nsswitch.conf`** files from the **`/usr/newconfig/etc/`** directory:

```
nsswitch.compat
nsswitch.files
nsswitch.hp_defaults
nsswitch.nis
nsswitch.nisplus
```

Note that the **`nsswitch.hp_defaults`** filename is a bit misleading—the policies shown in this file are NOT the default policies used in HP-UX 11.x! This file should be moved into place if you want your 11.x machine to use the same switch policy that was used by default at 10.x.

© 2003 Hewlett-Packard Development Company, L.P.

12–16. SLIDE: Testing Resolvers with `nsquery`



Student Notes

At HP-UX 11.x, you should use the **`nsquery`** command to test your resolver configuration:

```
# nsquery hosts sacramento.ca.hp.com
# nsquery hosts 128.1.1.4
```

The **`nsquery`** command first checks your **`/etc/nsswitch.conf`** file to determine which switch policy you have chosen to use. If you have chosen **`/etc/hosts`**, then **`nsquery`** simply searches the **`/etc/hosts`** file for the host name or IP address you have specified.

If you have chosen to use DNS as a lookup source, **`nsquery`** checks **`/etc/resolv.conf`** to find the address of your default name server, and forwards the resolution request accordingly. If the first name server times out, **`nsquery`** will try the second name server listed in **`/etc/resolv.conf`**. If none of the name servers in **`/etc/resolv.conf`** respond, **`nsquery`** displays a message indicating that the DNS lookup failed, then follows the "fallback" policy defined in your switch file to choose another lookup service.

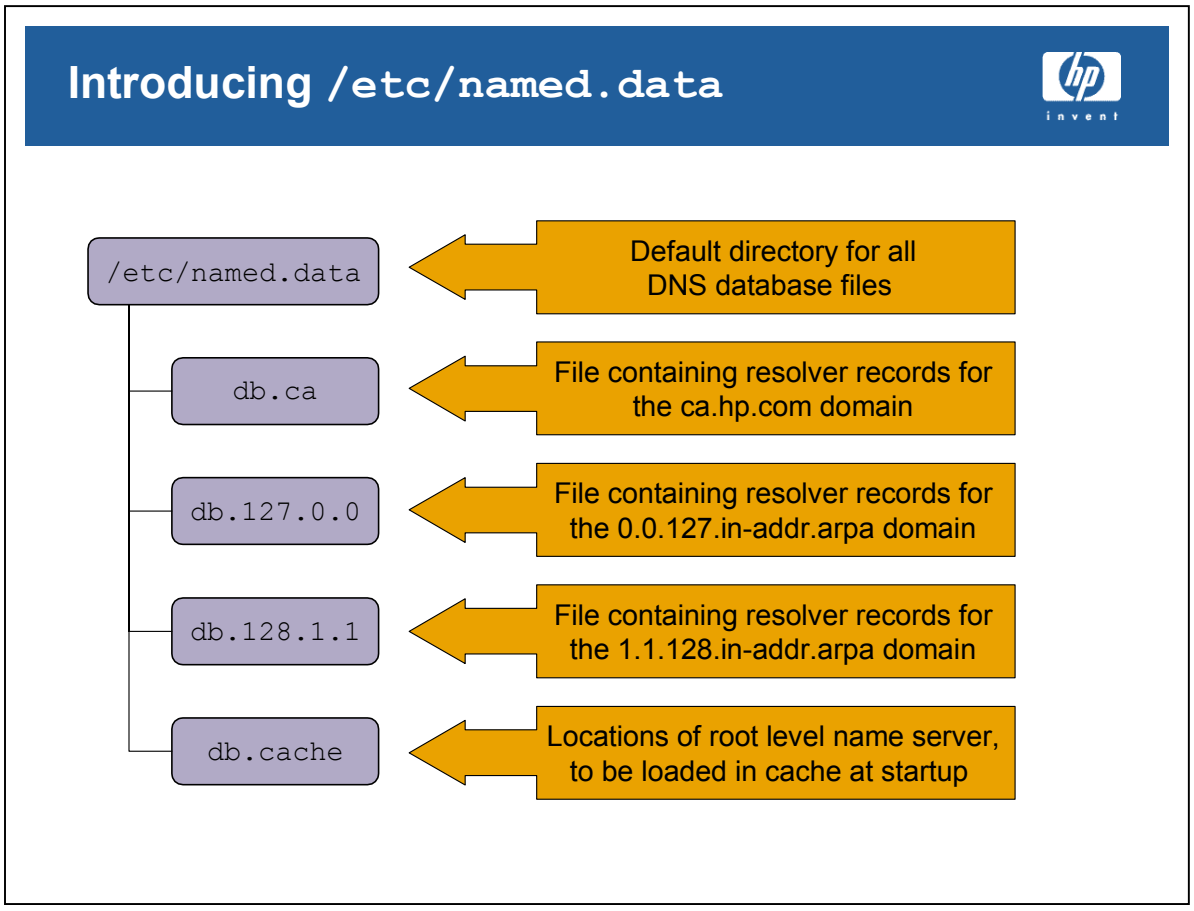
© 2003 Hewlett-Packard Development Company, L.P.

nsquery reports the result of each lookup service consulted, so you can determine if your switch policy behaves as expected.

CAUTION:	At HP-UX 10.20, use the **nslookup** command to test your resolvers. At HP-UX 11.0, **nslookup** was unable to interpret the **/etc/nsswitch.conf** file properly. The **nsquery** command is now the preferred command for testing the fallback resolution method defined in the **/etc/nsswitch.conf** file.

© 2003 Hewlett-Packard Development Company, L.P.

12–17. SLIDE: Introducing `/etc/named.data`



Student Notes

DNS name servers store their zone configuration data in a series of files under the **`/etc/named.data`** directory. This directory should contain one file for each of the domains for which your name server is authoritative source. The master name server for the **`ca.hp.com`** domain would have the following files in **`/etc/named.data`**:

`db.ca` Contains hostname to IP translation information for hosts in the **`ca.hp.com`** domain. Servers that are responsible for multiple domains have a separate **`db.`*`domain`*** file for each domain.

`db.127.0.0` Contains IP to hostname translation information for the loopback address in the **`0.0.127.in-addr.arpa`** domain.

`db.128.1.1` Contains IP to hostname translation information for the 128.1.1 subnet addresses in the **`1.1.128.in-addr.arpa`** domain. Servers for domains that span multiple subnets have a separate **`db.`*`x.x.x`*** file for each subnet.

`db.cache` Contains the addresses of the root level name servers, which are used for recursive queries. Some administrators mistakenly believe that this file

© 2003 Hewlett-Packard Development Company, L.P.

may be modified to force non-root-server addresses into cache. Not so. This file should only contain root-level server addresses.

`db.root` (Not shown on slide) This file replaces the **`db.cache`** file on root level name servers.

All of these are ASCII files that can be viewed directly and modified. For more information about the file contents, attend HP's DNS course (Course #H3540) or buy a copy of Cricket Liu's *DNS and BIND, Third Edition*, from O'Reilly and Associates (ISBN 1-56592-512-2).

CAUTION: The **`hosts_to_named`** utility overwrites the **`/etc/named.data/db.*`** files. If you modify any of the **`db.*`** files manually, do not run **`hosts_to_named`**!

© 2003 Hewlett-Packard Development Company, L.P.

12–18. SLIDE: Introducing `/etc/named.conf`

Introducing `/etc/named.conf`

`/etc/named.conf` on the master `ca.hp.com` name server:

```
// Define the DNS data directory

options                     {
                             check-names response fail;
                             check-names slave warn
                             directory = "/etc/named.data";
                            }

// Define which domains this name server
// can serve, and which file contains the
// records for each of those domains.  Note
// this name server is primary for all
// of the domains listed here.

zone "ca.hp.com"            { type master; file "db.ca";      };
zone "0.0.127.IN-ADDR.ARPA" { type master; file "db.127.0.0"; };
zone "1.1.128.IN-ADDR.ARPA" { type master; file "db.128.1.1"; };
zone "."                    { type hint;   file "db.cache";   };
```

Student Notes

When the **named** daemon is launched during system startup, it consults a file called **/etc/named.conf** to determine which domains it is responsible for, and which **db.*** files need to be loaded. The slide shows the **/etc/named.conf** file on **sanfran**, the master name server for the **ca.hp.com** domain.

The **options** block at the top of the file defines some general parameters for the daemon. In the example on the slide, the two **check-names** directives cause **named** to verify the format hostnames that this server obtains via recursive queries to other servers. If a recursive query yields a hostname that contains an underscore or other non-standard characters, **named** will refuse to send the results back to the client that requested the lookup. This directive is designed to prevent syntax errors in other servers' database files from filtering back to your resolver clients.

The **directory** directive tells **named** in which directory the **db.*** files are stored.

The remaining lines in the sample file tell **named** for which zones it is responsible. Each line has several fields. The **zone** directive specifies a zone name. The **type** directive indicates whether the server is a master or slave for the zone. The **file** directive specifies the name of

© 2003 Hewlett-Packard Development Company, L.P.

the database file containing the zone information. Slave servers have one more field with each record: a **master** directive that specifies the IP address of the master server that the slave should query for regular updates.

Many more options are available in the **named.conf** file. See the previously mentioned O'Reilly DNS book, or read the **named** man page for more information.

Sample /etc/named.conf File on a Slave Server

The sample file below was taken from a slave server in the **ca.hp.com** domain. How does this file differ from the master server sample file on the slide?

```
options { check-names response fail;
          check-names slave    warn;
          directory "/etc/named.data";             };

zone "0.0.127.IN-ADDR.ARPA" { type master;
                              file "db.127.0.0";   };
zone "1.1.128.IN-ADDR.ARPA" { type slave;
                              file "db.128.1.1";
                              masters { 128.1.1.1; }; };
zone "ca.hp.com"            { type slave;
                              file "db.ca";
                              masters { 128.1.1.1; }; };
zone "."                    { type hint;
                              file "db.cache";     };
```

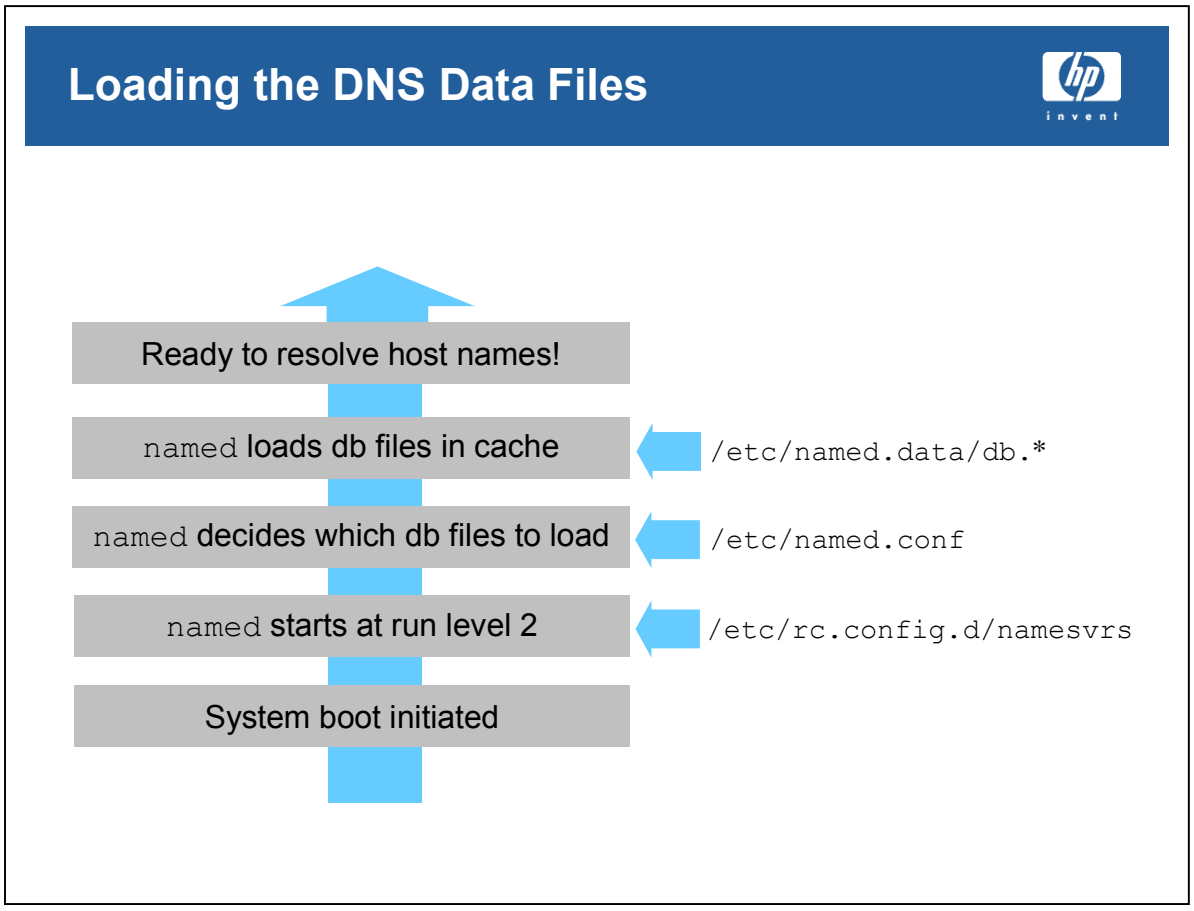
Sample /etc/named.conf File on a Cache-Only Server

The sample file below was taken from a cache-only server in the **ca.hp.com** domain. How does this file differ from the master server sample file on the slide?

```
options { check-names response fail;
          check-names slave    warn;
          directory "/etc/named.data";               };

zone "0.0.127.in-addr.arpa" { type master;
                              file "db.127.0.0";   };
zone "."                    { type hint;
                              file "db.cache";     };
```


© 2003 Hewlett-Packard Development Company, L.P.

12–19. SLIDE: Loading the DNS Data Files



Student Notes

When the system boots to run level 2 or higher, the **/sbin/init.d/named** searches in the **/etc/rc.config.d/namesvrs** file and starts the **named** daemon if the **NAMED** control variable is set to 1.

The **named** daemon reads **/etc/named.conf** to determine the zones for which it is responsible, then reads in the appropriate **/etc/named.data/db.*** files into memory. Note that **named** reads only the DNS database files at startup. If you make any changes to the **db.*** files, you must force **named** to re-read its database files as described on the next slide.

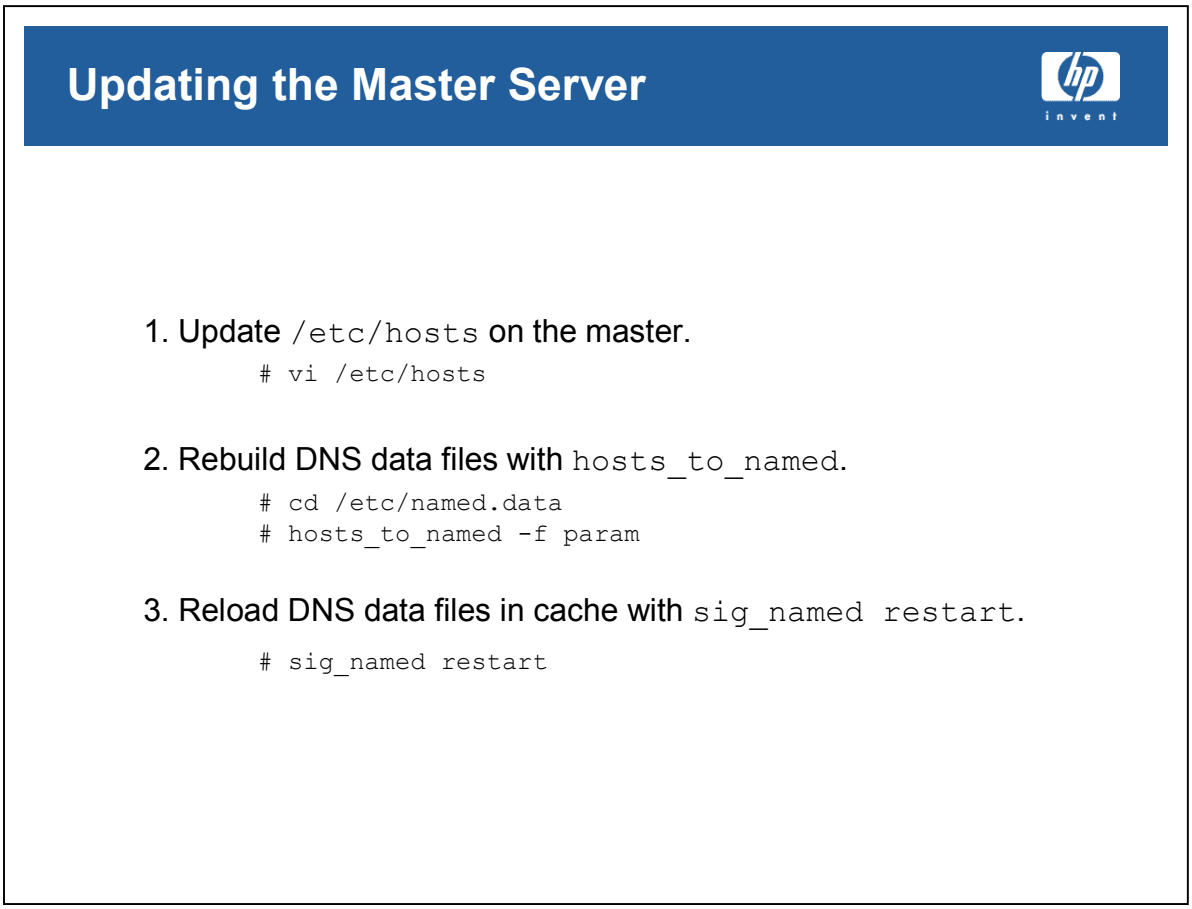
You can stop or start **named** by executing the startup script with the appropriate argument:

```
# /sbin/init.d/named stop
# /sbin/init.d/named start
```

NOTE: **named** runs only on DNS servers, not on resolver-only clients.

© 2003 Hewlett-Packard Development Company, L.P.

12–20. SLIDE: Updating the Master Server



Student Notes

Any time a hostname or IP address is added, removed, or changed in your DNS domain, the name server data files must be updated accordingly. You could make these changes directly with **vi**, but in smaller domains, it is often easier to update **/etc/hosts**, then rerun **hosts_to_named**.

The example below adds a host named "sacramento" with IP address 128.1.1.4 to the **ca.hp.com** domain.

1. Update **/etc/hosts** on the master server.

   Add a new line to **/etc/hosts** for each new host name/IP pair. Be sure to use fully qualified host names.

```
# vi /etc/hosts
   127.0.0.1       localhost
   128.1.1.1       sanfran.ca.hp.com.      sanfran
   128.1.1.2       oakland.ca.hp.com.      oakland
   128.1.1.3       la.ca.hp.com.           la
   128.1.1.4       sacramento.ca.hp.com.   sacramento
```


© 2003 Hewlett-Packard Development Company, L.P.

2. Rerun **hosts_to_named** on the master server.

   This will rebuild the master server's DNS data files to reflect the changes made in **/etc/hosts.**

   ```
   # cd /etc/named.data
   # hosts_to_named -f param
   ```

3. Run **sig_named** on the master.

   By default, **named** only reads the **db** files at startup. The **sig_named** command forces the **named** daemon on the master to reload any updated database files.

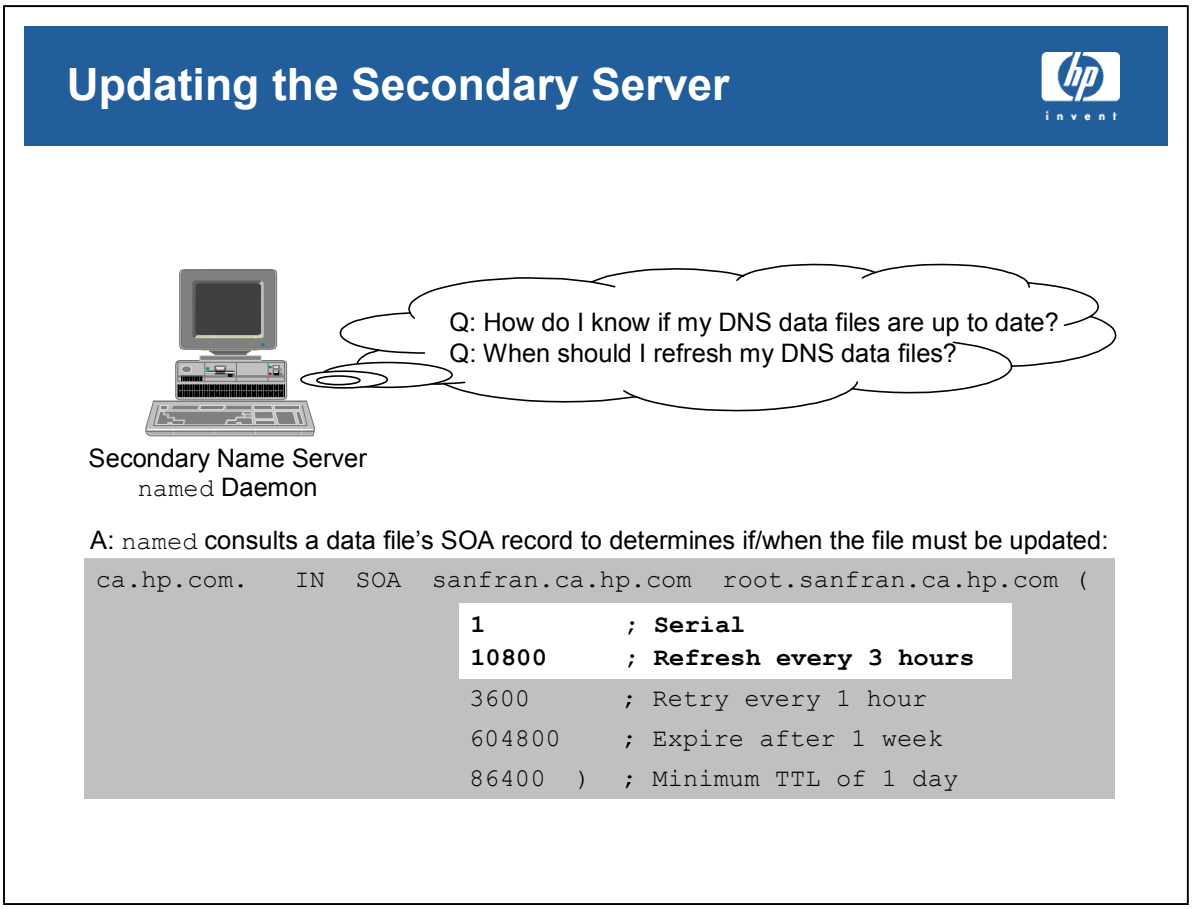
   ```
   # sig_named restart
   ```

Note that the slave servers will not be updated immediately. Turn to the next slide to learn how the slave server data files are updated.

CAUTION: The **hosts_to_named** utility overwrites the **/etc/named.data/db.*** files. If you modify any of the **db.*** files manually, do not run **hosts_to_named**!

© 2003 Hewlett-Packard Development Company, L.P.

12–21. SLIDE: Updating the Secondary Server



Student Notes

When hostname and IP address changes are required, the changes are made on the DNS master server. Every slave server should be configured to periodically query the master server to determine if an update is required.

Every DNS database file has a "Start of Authority" (SOA) record at the top of the file that determines how frequently slave servers request updates from their master servers. Consider the sample start of authority record on the slide.

The first line in the SOA identifies the domain name (**ca.hp.com**) and master server name (**sanfran.ca.hp.com**), and the domain administrator's email address (**root.sanfran.ca.hp.com** = **root@sanfran.ca.hp.com**).

The remaining fields determine how frequently the zone updates occur:

Serial Each zone has a serial number. Slave servers determine if their database files are up-to-date by comparing their zone data file serial numbers against the serial numbers on the master's data files. If the master's number is greater than the slave's, the slave requests a zone transfer. The master server

© 2003 Hewlett-Packard Development Company, L.P.

administrator must remember to increment the serial number in the SOA any time a **db.*** file is modified (**hosts_to_named** does this automatically).

Refresh This field determines how frequently slave servers should request updates from the master. The interval is specified in seconds.

Retry If the master does not respond to a slave's update request, the **Retry** field determines how long the slave should wait before trying again. This parameter, too, is defined in seconds.

Expire If one week passes without a successful update from the master, the slave shown on the slide expires the zone data and refuses to answer client queries about the expired zone. This parameter, too, is defined in seconds.

TTL The "Time To Live" determines how long other name servers (not slave servers) may retain this zone data in cache. This parameter, too, is defined in seconds.

If you want to force an immediate zone transfer on your slave server, execute the **sig_named** restart command. Note that there is no mechanism in DNS that allows the master to "push" an immediate zone transfer to the slaves; slaves are expected to "pull" updates at regular intervals.

NOTE: Slave servers update themselves automatically. An immediate update can be performed at any time by manually executing **sig_named restart** on the slave server.

© 2003 Hewlett-Packard Development Company, L.P.

12–22. LAB: Configuring DNS

Introduction

In this exercise, you will configure a DNS master server, a slave server, and a DNS client. You will also have a chance to update the DNS data on your name servers, and explore some of the name server database files.

Your instructor will break the class into teams of 2 or 3 students each. Each team will be assigned a DNS sub-domain under **`hp.com`** from the table below. You will then work with your teammates to configure a master server, a slave server, and one or more DNS clients within your assigned domain. The instructor's station will serve as a root level name server so you can access other teams' domains as well.

Table 12-1.

Domain Name	Role	Host Name	IP Address
`hp.com`	master	corp	128.1.0.1
`ca.hp.com`	master	sanfran	128.1.1.1
	slave	oakland	128.1.1.2
	client	la	128.1.1.3
`il.hp.com`	master	chicago	128.1.2.1
	slave	peoria	128.1.2.2
	client	rockford	128.1.2.3
`ga.hp.com`	master	atlanta	128.1.3.1
	slave	athens	128.1.3.2
	client	macon	128.1.3.3
`ny.hp.com`	master	nyc	128.1.4.1
	slave	albany	128.1.4.2
	client	buffalo	128.1.4.3
`fr.hp.com`	master	paris	128.1.5.1
	slave	lyon	128.1.5.2
	client	grenoble	128.1.5.3
`uk.hp.com`	master	london	128.1.6.1
	slave	leeds	128.1.6.2
	client	ipswich	128.1.6.3
`de.hp.com`	master	bonn	128.1.7.1
	slave	berlin	128.1.7.2
	client	hamburg	128.1.7.3
`jp.hp.com`	master	tokyo	128.1.8.1
	slave	kyoto	128.1.8.2
	client	osaka	128.1.8.3

© 2003 Hewlett-Packard Development Company, L.P.

Part 1: Configure Your Master Server

1. Delete all entries from the **/etc/hosts** file except the **localhost** entry and the hosts in your domain. Fully qualify all of the host names. The example below shows the changes that are required on **sanfran**.

```
# vi /etc/hosts
127.0.0.1 localhost
128.1.1.1 sanfran.ca.hp.com sanfran
128.1.1.2 oakland.ca.hp.com oakland
128.1.1.3 la.ca.hp.com      la
```

2. Create a directory for the DNS database files and **cd** to it.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
# cd /etc/named.data
```

3. Create a **param** file for your domain.

```
# vi param
    -d ca.hp.com        # Use your domain name(s) here
    -n 128.1.1          # Use your subnet address(es) here
    -z 128.1.1.1        # Use your master server's IP here
    -b /etc/named.conf
```

4. Run **hosts_to_named**.

```
# hosts_to_named -f param
```

5. Copy the **db.cache** file from the instructor station.

```
# ftp 128.1.0.1
> get /etc/named.data/db.cache
> quit
```

6. Enable **NAMED** in the **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

7. Start the **named** daemon.

```
# /sbin/init.d/named start
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configure Your Slave Server

1. Create a directory for the database and configuration files and **cd** into it.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
# cd /etc/named.data
```

2. **ftp** copies of the **db.*** files from the master.

```
# ftp 128.1.1.1      # Use your master server's IP here
> mget /etc/named.data/db.*
> quit
```

3. FTP a copy of **conf.sec.save** from the master server, and move it into place on the slave server as **/etc/named.conf**.

```
# ftp 128.1.1.1       # Use your master server's IP here
> get /etc/named.data/conf.sec.save
> quit
# mv /etc/named.data/conf.sec.save /etc/named.conf
```

4. Enable **NAMED** in **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

5. Start the **named** daemon. A reboot is not necessary.

```
# /sbin/init.d/named start
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configure All Hosts (Including Your DNS Servers) in Your Domain as DNS Clients

1. Create/modify the resolver configuration file. Include your domain and the **hp.com** domain in your **search** list. Include both your master and your slave server in the **nameserver** list.

```
# vi /etc/resolv.conf
  search ca.hp.com hp.com # replace ca.hp.com with your domain name
  nameserver 128.1.1.1    # replace 128.1.1.1 with your master's IP
  nameserver 128.1.1.2    # replace 128.1.1.2 with your slave's IP
```

2. If your **/etc/nsswitch.conf** exists, delete it. You can experiment with the default behavior for now. You will have a chance to re-create the file later.

```
# rm /etc/nsswitch.conf
```

3. If you are the master server, skip this step! Slaves and clients need to modify **/etc/hosts** at this point. Fully qualify and create aliases for your host in your local domain, and remove all other entries (except **localhost**).

```
# vi /etc/hosts
     127.0.0.1   localhost
     128.1.1.3   la.ca.hp.com  la
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Test DNS

All hosts in your domain (clients and servers) can try the following exercises.

1. Run **`nslookup`** and identify your master server as the server to use.

   - Can you resolve a host name in your own domain?
   - Can you resolve an IP address in your own domain?
   - Can you resolve a host name in another domain? (Try **`corp.hp.com`**.)
   - Can you resolve an IP address in another domain? (Try **`128.1.0.1`**.)

2. Try the same tests that you did in the previous question, but use the slave server this time. Does your slave server seem to work?

3. Which name server does **`nslookup`** use by default if you simply type **`nslookup corp.hp.com`** from the shell prompt? Try it. How can you permanently change the default name server?

4. Try resolving a host name in your domain using the simple host name (that is, **`sanfran`**, rather than **`sanfran.ca.hp.com`**). Try resolving a host in another domain using the simple host name. Your first experiment should succeed, while the second should fail. Why?

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Updating Your DNS Name Servers

1. Choose a new host/IP for your domain, and add it to your master server's DNS data files using vi and **`hosts_to_named`**. Do not run **`sig_named`** yet. Note that you can add a new host name/IP to DNS even if that host has not been physically connected to the network yet.

2. Which two **`db.*`** files would you expect to be affected by the newly added host and IP? Look at the SOA records for those two files. How can you tell that the files were updated?

3. Now that the **`db.*`** files have been updated, can you **`nslookup`** the new host using the master server? Try it, and explain the results.

4. What do you need to do to ensure that your DNS clients can resolve the new host name? Make it so.

5. By default, when will your slave server recognize that a new host name and IP have been added to the domain? How can you force the slave to do an immediate update? Do it.

6. Verify that the slave server update was successful.

```
# nslookup
> server 128.1.1.2          ← Use your slave server's IP here.
> sacramento.ca.hp.com      ← This should succeed!
> exit
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Handling DNS Server Failures

1. This portion of the lab exercise asks you to take your LAN card down, and bring it back up again several times. Losing LAN connectivity will generally cause CDE to hang. Before attempting this portion of the lab exercise, shutdown CDE:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. What happens if the master DNS server becomes unreachable? Can clients still resolve host names? Use the **`ifconfig`** command to take down the LAN card on your master server and see if your clients can still resolve **`corp.hp.com`**. (Be patient!)

   ```
   master# ifconfig lan0 down
   client# nsquery hosts corp.hp.com
   ```

   How is the client able to resolve host name **`corp`** if the master server is inaccessible?

3. What happens if the slave server becomes unavailable, too? Try it. While the LAN card on the master server is still down, disable the LAN card on the slave server, too. Then attempt to resolve **`corp.hp.com`** again.

   ```
   master# ifconfig lan0 down
    slave# ifconfig lan0 down
   client# nsquery hosts corp.hp.com
   ```

   What happens this time?

4. Bring both servers' LAN cards back up again.

   ```
   master# ifconfig lan0 up
    slave# ifconfig lan0 up
   ```


© 2003 Hewlett-Packard Development Company, L.P.

5. Occasionally, you may have some development machines or other temporary hosts on your LAN. Rather than add these temporary host names to your DNS server databases, you may wish simply to record them in the local hosts file. Add a new host name/IP pair to your client's local hosts file. Then **`nsquery`** the new host name. What happens? Why?

6. Change the client's hosts policy in **`/etc/nsswitch.conf`**. Configure your client's resolver to check the local **`/etc/hosts`** if DNS is unable to resolve a host name. Then **`nsquery`** the new host name you added to your client's **`/etc/hosts`** file in the previous step. What happens?

Part 7: Cleanup

1. Restore your pre-DNS configuration on all hosts in your domain by running **`netfiles.sh`**:

```
master# /labs/netfiles.sh -r NEW
 slave# /labs/netfiles.sh -r NEW
client# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

Module 13 — Configuring the ARPA/Berkeley Services

Objectives

Upon completion of this module, you will be able to do the following:

- List the commonly used ARPA-Berkeley services.
- Describe the function of the Internet daemon, **`inetd`**.
- Describe the process used to request **`ftp/telnet`** service from **`inetd`**.
- Describe the Internet service configuration files.
- Enable or disable Internet services from the command line.
- Allow or prevent access to selected Internet services via the **`inetd.conf`** file.
- Allow/prevent access for selected clients via the **`inetd.sec`** file.
- Allow/prevent access for selected users via the **`passwd`** file.
- Log requests for ARPA/Berkeley services.
- Define host equivalency between hosts with the **`/etc/hosts.equiv`** file.
- Define user equivalency between hosts with the **`~/.rhosts`** file.

© 2003 Hewlett-Packard Development Company, L.P.

13–1. SLIDE: Internet Services Overview

Internet Services Overview

Capability	ARPA	Berkeley
Terminal access	`telnet`	`rlogin`
File transfer	`ftp`, `tftp`	`rcp`
Remote command execution		`remsh`, `rexec`
Electronic mail	SMTP	`sendmail` (uses SMTP)
Interprocess communication		Sockets
Network information	`finger`	`rwho`, `ruptime`
Dynamic routing	`gated`	
Name service		BIND
Time synchronization	NTP	
Remote boot	BOOTP	
Remote printing		`printer` (`rlpdaemon`)

Student Notes

The Internet Services are among the most frequently used network applications. The HP-UX Internet Services product includes utilities for remotely logging into other hosts on the LAN, transferring files across the LAN, delivering email, and many other basic services.

The Internet Services product includes two families of utilities: the ARPA services and Berkeley services. The chart on the slide and the notes below overview some of the features these services provide.

© 2003 Hewlett-Packard Development Company, L.P.

ARPA Services

ARPA services are the de facto networking standards in the scientific and engineering communities. For LANs and WANs, they define protocols for:

- terminal access (**`telnet`**)
- file transfer (**`ftp`**, the file transfer protocol, and **`tftp`**, the trivial file transfer protocol)
- electronic mail (SMTP, the simple mail transfer protocol)
- dynamic routing (**`gated`** supports several routing protocols)
- time synchronization (NTP, the network time protocol)
- remote booting (**`bootp`**), used by X stations and NFS diskless systems

ARPA services are available on different operating systems, such as HP-UX, other UNIX systems, RTE-A, MPE/iX, MS-DOS, and VMS.

Berkeley Services

BSD UNIX 4.3 implements a de facto networking standard for the UNIX community.

For LANs and WANs, it defines protocols for

- terminal access (**`rlogin`**)
- file transfer (**`rcp`**)
- remote command execution (**`remsh`**, **`rexec`**)
- electronic mail (**`sendmail`**)
- interprocess communication (Berkeley Sockets API)
- getting network information (**`rwho`**, **`ruptime`**, **`finger`**)
- mapping host names to IP addresses (BIND DNS, the BIND Domain Name Service)
- remote printing (**`rlp`** daemon)

© 2003 Hewlett-Packard Development Company, L.P.

The Internet Services can be put in the context of the OSI model as shown.

OSI Model	ARPA Services	Berkeley Services	Product Structure
7 Application	ftp telnet bootp tftp named gated xntpd	rcp rlogin remsh rexec rwho ruptime sendmail printer	Services
6 Presentation	SMTP		
5 Session		BSD IPC	
4 Transport	TCP	TCP	LAN Link
3 Network	IP	IP	
2 Data Link	Ethernet	Ethernet	
1 Physical	Ethernet/ IEEE 802.3	Ethernet/ IEEE 802.3	

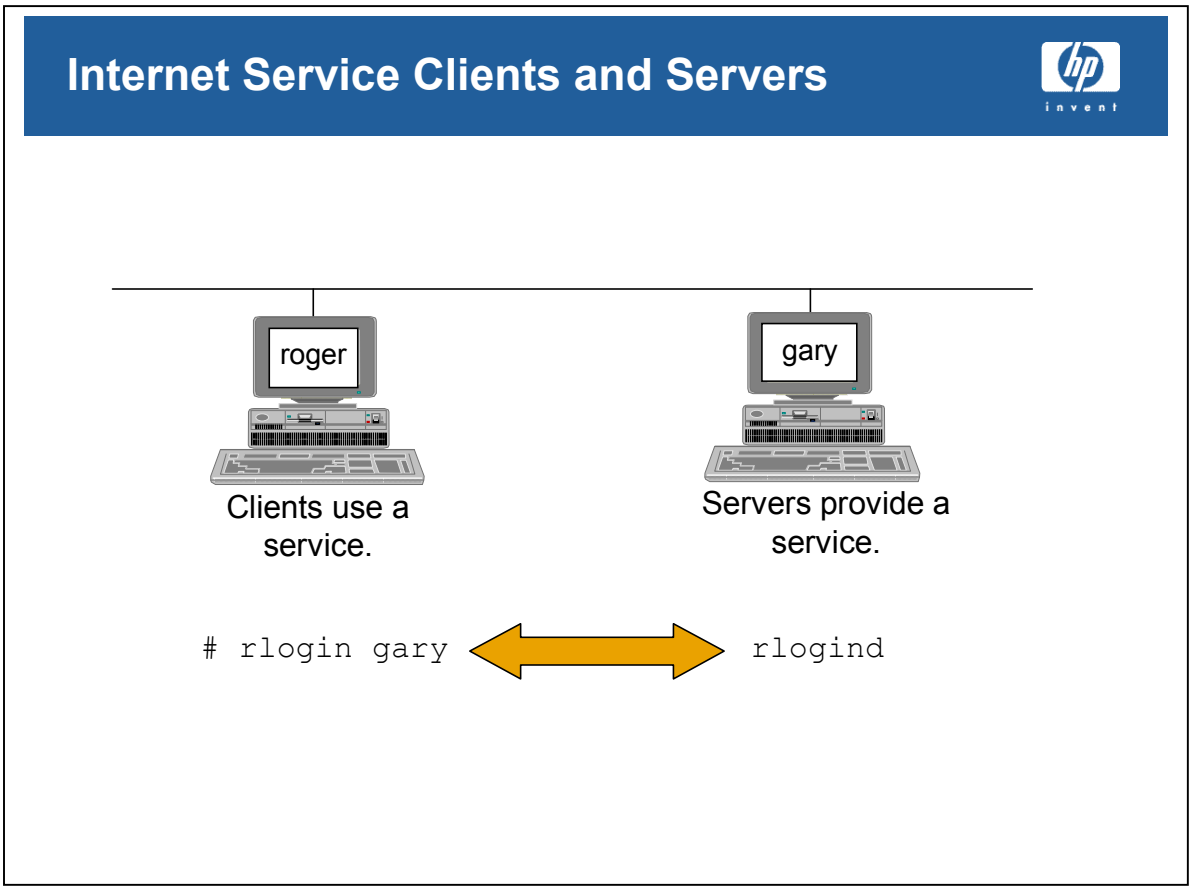
Figure 1

NOTE: The Internet Services software product requires the LAN/9000 Link, FDDI 9000/Link, Token Ring/9000 Link or X.25 Link product.

The **`sendmail`** utility, dynamic routing with gated, BIND, and time synchronization with NTP will not be discussed in this module.

© 2003 Hewlett-Packard Development Company, L.P.

13–2. SLIDE: Internet Service Clients and Servers



Student Notes

The Internet Services are built on a client-server model.

A **client** uses services that a **server** provides. The term **client/server** is very often used with systems and not with processes, but a server system can provide a service only when a server process is running there. On the other side a client system can only use a service when its client process is able to communicate with the appropriate server process on the server system.

A system can be simultaneously a server and a client if server processes and as well client processes are running there.

The slide shows a very simple example of a client/server relationship. A user executes the **`rlogin`** command on node **`roger`** to get a virtual terminal on the remote node **`gary`**. The **`rlogin`** program is the client process. The appropriate server process, **`rlogind`**, is then invoked on node **`gary`**, and a network communication session is established between **`rlogin`** and **`rlogind`**.

© 2003 Hewlett-Packard Development Company, L.P.

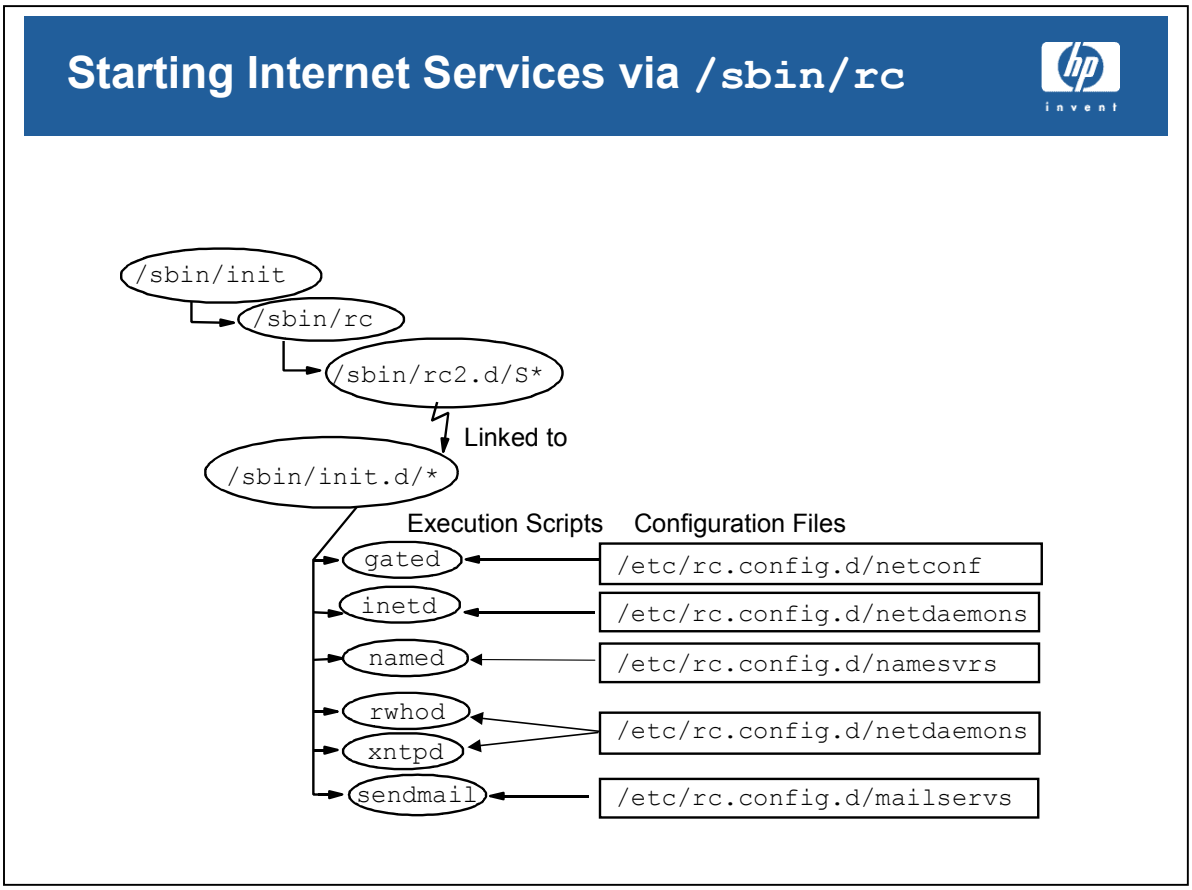
The following table shows other client/server relationships within the Internet Services:

Table 1

Service	Client	Server
Terminal access	**telnet**	**telnetd**
	rlogin	**rlogind**
File transfer	**ftp**	**ftpd**
	rcp	**remshd**
Remote command execution	**remsh**	**remshd**
	rexec	**rexecd**
Network information	**finger**	**fingerd**
	rup, **ruptime**	**rwhod**

© 2003 Hewlett-Packard Development Company, L.P.

13–3. SLIDE: Starting Internet Services via `/sbin/rc`



Student Notes

Many of the Internet Services have server daemons that are started at run-level 2 during the boot process, and run continuously on the system.

Internet services that have dedicated server daemons include:

- `gated`
- `named`
- `rwhod`
- `xntpd`
- `sendmail`

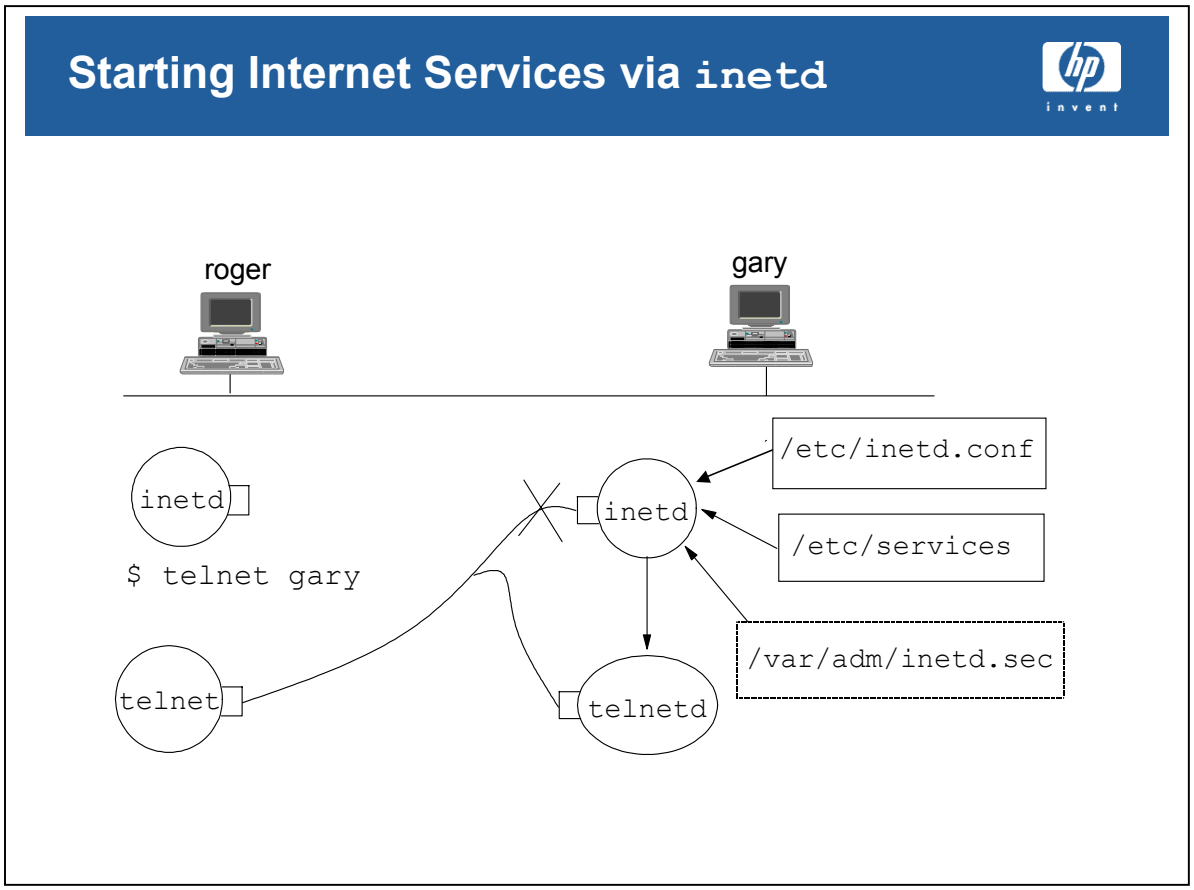
Each of these services has a startup/shutdown script in **`/sbin/init.d`**, and an associated configuration script in the **`/etc/rc.config.d`** directory.

Some of these services may be disabled. Be sure to check the control variables in the **`/etc/rc.config.d`** files (especially **`netdaemons`**), to determine which services are enabled and which are disabled on your system.

Server processes for the remaining Internet services that are not included in the list above are all managed by the **`inetd`** "superdaemon" which is introduced on the next slide.

© 2003 Hewlett-Packard Development Company, L.P.

13–4. SLIDE: Starting Internet Services via `inetd`



Student Notes

Although many of the internet services have daemons that run continuously on the system, some internet service server processes are managed by the **`inetd`** "super-daemon."

The **`inetd`** daemon starts at run-level 2 during the system boot process, and monitors the server's ports for requests for a variety of internet services. When a client requests access to one of the services provided by **`inetd`**, **`inetd`** starts whatever server process is necessary to respond to the client's request. The server process handles all further communication with the client so **`inetd`** can listen for additional service requests.

Internet services managed by the **`inetd`** super-daemon include:

- **`telnet`**
- **`ftp`**
- **`tftp`**
- **`bootp`**
- **`rlogin`**
- **`remsh`**
- And many others

© 2003 Hewlett-Packard Development Company, L.P.

Starting server processes via **inetd** offers two major advantages. First, since server processes are only started on an as-needed basis, the system load on the server is reduced. Second, **inetd** makes it possible for the server to maintain connections to multiple clients simultaneously. The **inetd** daemon simply starts an additional server process for each additional client. Thus, if three clients **telnet** to your server, **inetd** will start three **telnetd** server processes.

NOTE:	The **inetd** daemon is only needed on the server side. You should be able to **telnet** and **ftp** out to other hosts even if **inetd** is not running.

The **inetd** daemon starts at run-level 2 and runs continuously on the system until shutdown. Unlike most other scripts executed during the boot process, **/sbin/init.d/inetd** does not have a control variable. Thus, if you do not want to start **inetd** at boot, you must remove the **inetd** start script from **/sbin/rc2.d**.

You can manually stop or start **inetd** by executing the **inetd** startup script:

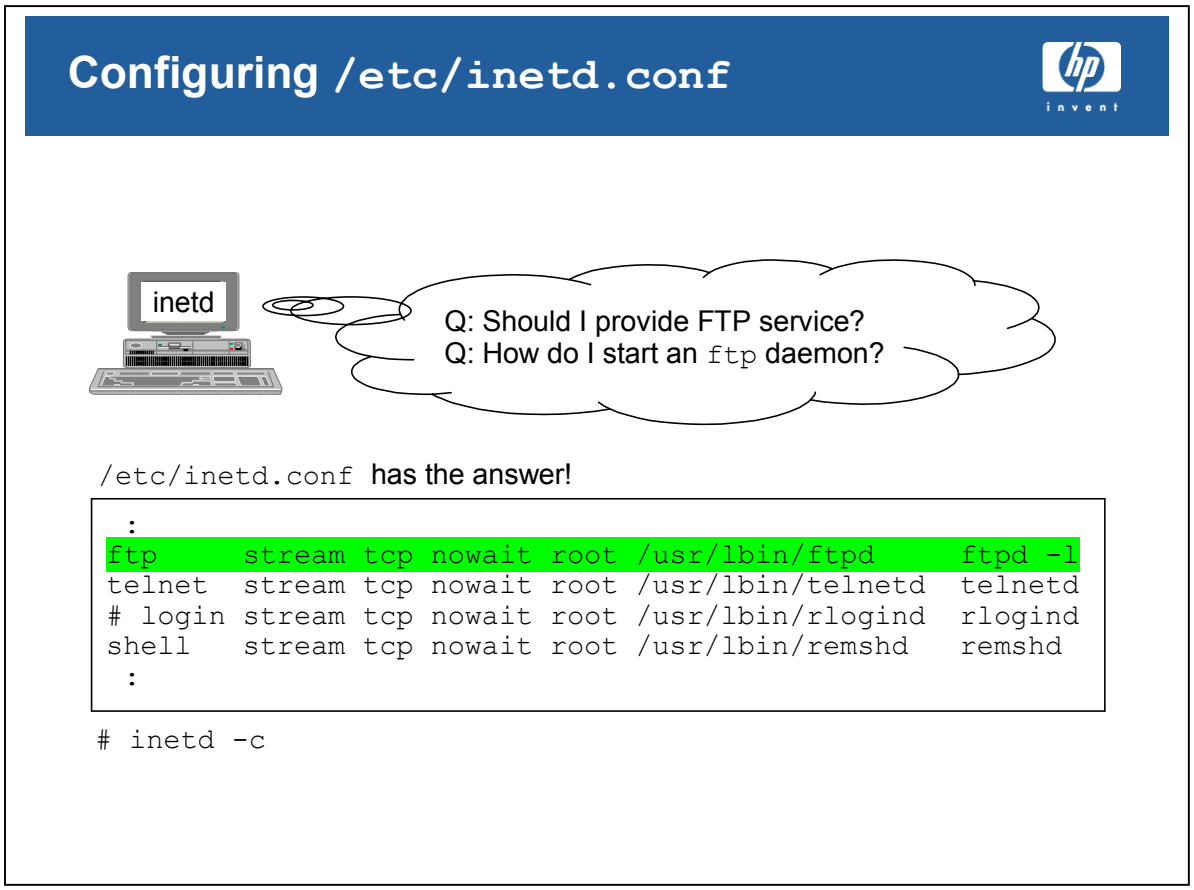
```
# /sbin/init.d/inetd stop
# /sbin/init.d/inetd start
```

The **inetd** daemon references several configuration files that are described in the slides that follow:

```
/etc/inetd.conf
/etc/services
/var/adm/inetd.sec
```


© 2003 Hewlett-Packard Development Company, L.P.

13–5. SLIDE: Configuring `/etc/inetd.conf`



Student Notes

When **`inetd`** is invoked, it reads the **`/etc/inetd.conf`** configuration file and configures itself to support whatever services are included in the file.

To disable an incoming service, you can use the comment sign **#** in /**`etc/inetd.conf`**.

NOTE:	If you modify the **`/etc/inetd.conf`** file, you have to force **`inetd`** to reread its configuration file. Use **`inetd -c`**.

The following are the fields in the **`/etc/inetd.conf`** file:

service name	The name of a valid service in the file **`/etc/services`** or, if the server is RPC-based (**`nfs`**), the service name should be in **`rpc`**.
socket type	Either **`stream`** or **`dgram`**, depending on whether the server socket is a stream or a datagram socket. Sockets will be discussed later in this module.
protocol	Must be a valid protocol as defined in **`/etc/protocols`**; for example, **`tcp`** or **`udp`**.

© 2003 Hewlett-Packard Development Company, L.P.

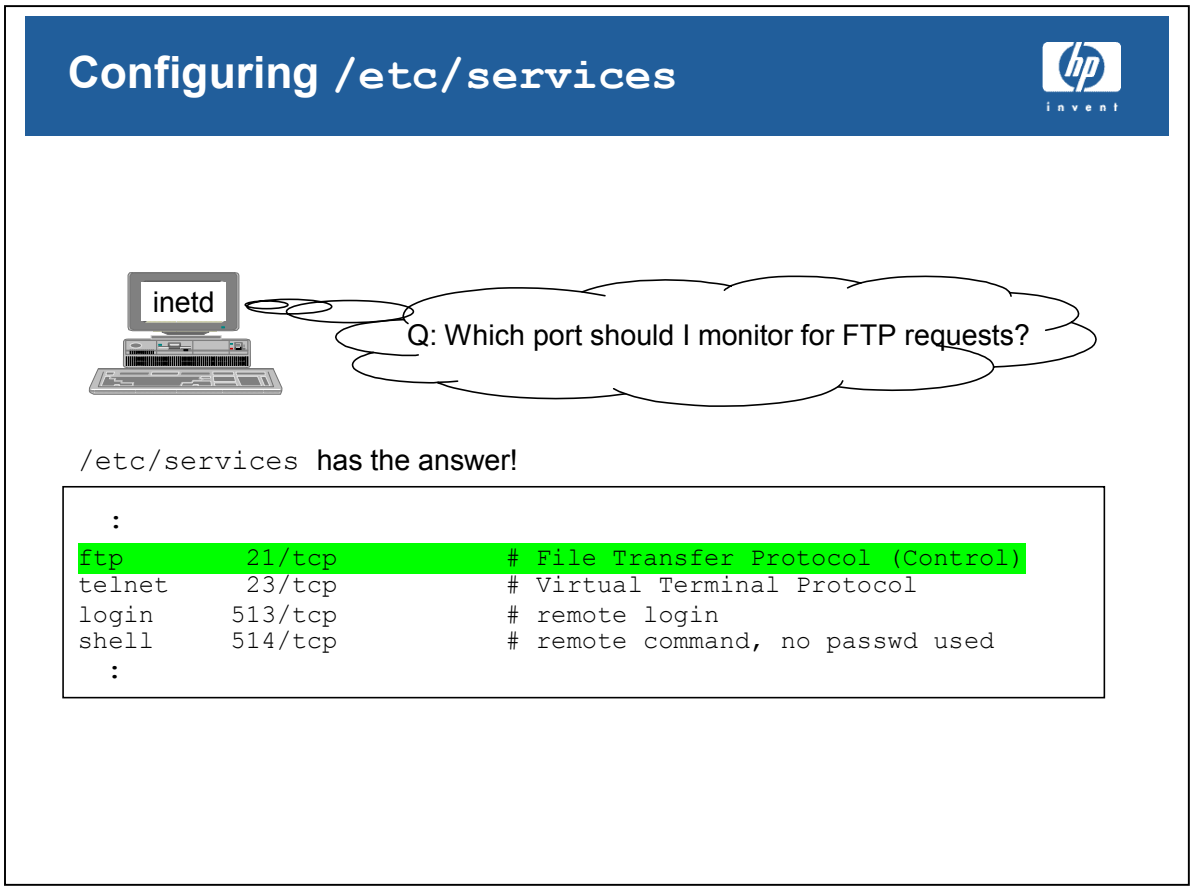
wait/nowait	**`wait`** applies to datagram sockets only. All other sockets should specify **`nowait. wait`** instructs **`inetd`** to execute only one datagram server for the specified socket at any one time. It instructs **`inetd`** to execute a datagram server for a specified socket whenever a datagram arrives.
user	The name of the user as whom the server should run.
server program	The absolute path name of the program which **`inetd`** executes when it finds a request on the server's socket.
arguments	The arguments to the server program starting with **`argv[0]`**, which is the name of the program.

An Example `/etc/inetd.conf` File

```
            :
            :
##
#
#       ARPA/Berkeley services
#
##
ftp          stream tcp nowait root /usr/lbin/ftpd ftpd -l
telnet       stream tcp nowait root /usr/lbin/telnetd telnetd
# Before uncommenting the "tftp" entry below, please make sure
# that you have a "tftp" user in /etc/passwd. If you don't
# have one, please consult the tftpd(1M) manual entry for
# information about setting up this service.

tftp       dgram   udp   wait   root   /usr/lbin/tftpd  tftpd\
     /usr/lib/sw/HP-UX.install
#bootps    dgram   udp   wait   root   /usr/lbin/bootpd bootpd
#finger    stream  tcp   nowait bin    /usr/lbin/fingerd fingerd
login      stream  tcp   nowait root   /usr/lbin/rlogind rlogind
shell      stream  tcp   nowait root   /usr/lbin/remshd remshd
exec       stream  tcp   nowait root   /usr/lbin/rexecd rexecd
#uucp      stream  tcp   nowait root   /usr/sbin/uucpd uucpd
##
#
#       Other HP-UX network services
#
##
printer stream tcp nowait root /usr/sbin/rlpdaemon rlpdaemon -i
            :
            :
```


© 2003 Hewlett-Packard Development Company, L.P.

13–6. SLIDE: Configuring `/etc/services`



Student Notes

Recall that a packet's destination is determined by the packet's destination socket address. The socket address is a concatenation of the destination host's IP address, and a port number on the destination host. The socket address allows the system to deliver each packet to the appropriate destination.

Each internet service has a "well-known" port number that is consistent across all hosts. The **`/etc/services`** file associates these well-known port numbers with service names.

After reading **`/etc/inetd.conf`** to determine which services it should provide, **`inetd`** consults **`/etc/services`** to determine which ports it should monitor for client requests for those services.

Lines in **`/etc/services`** may be commented out with a "#" sign to prevent access to a particular service. However, the more conventional approach to disabling a service is to comment the service's line out of **`/etc/inetd.conf`**.

Establishing a Connection

Let's take a closer look at what occurs when a client attempts to connect to a server. The example considers the steps required to initiate a **`telnet`** connection between two hosts.

© 2003 Hewlett-Packard Development Company, L.P.

First, the **inetd** daemon is started automatically during system startup. After reading **/etc/inetd.conf** and **/etc/services**, **inetd** determines that it should listen for **telnet** requests on well-known port number 23. If other services are configured in **inetd.conf**, **inetd** listens for connection requests on those services' well-known ports, too.

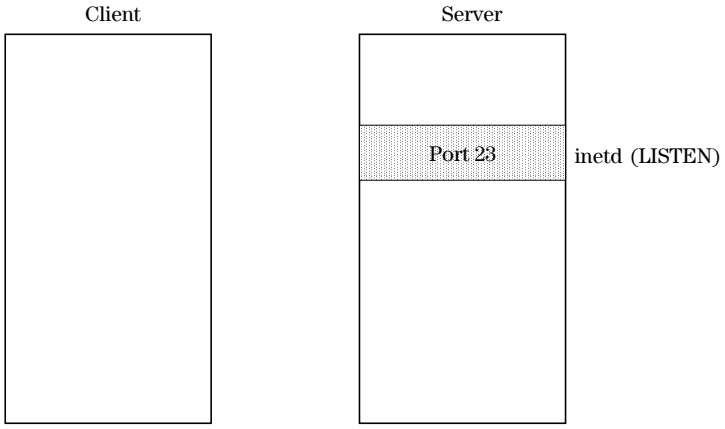


Figure 2

When a user on the client issues the **telnet** command, the **telnet** client process opens any available port on the client and sends a connection request to the well-known **telnet** port number 23 on the server. There is no need for the client **telnet** process to use a well-known port number, since nobody is trying to find the client process. Server processes, however, must use well-known port numbers so clients know which port to address their connection requests to.

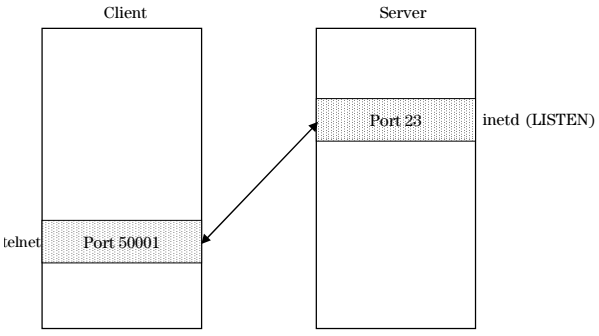


Figure 3

© 2003 Hewlett-Packard Development Company, L.P.

The server's **inetd** daemon receives the request for service on port 23. Since port 23 is the well-known port for **telnet**, **inetd** spawns a **telnetd** server process and establishes a socket connection upon which the **telnetd** and **telnet** processes communicate directly without intervention from **inetd**. **inetd** continues listening for new requests.

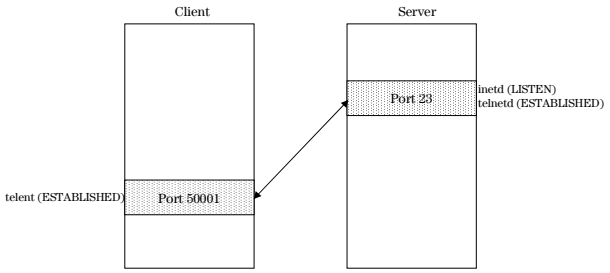


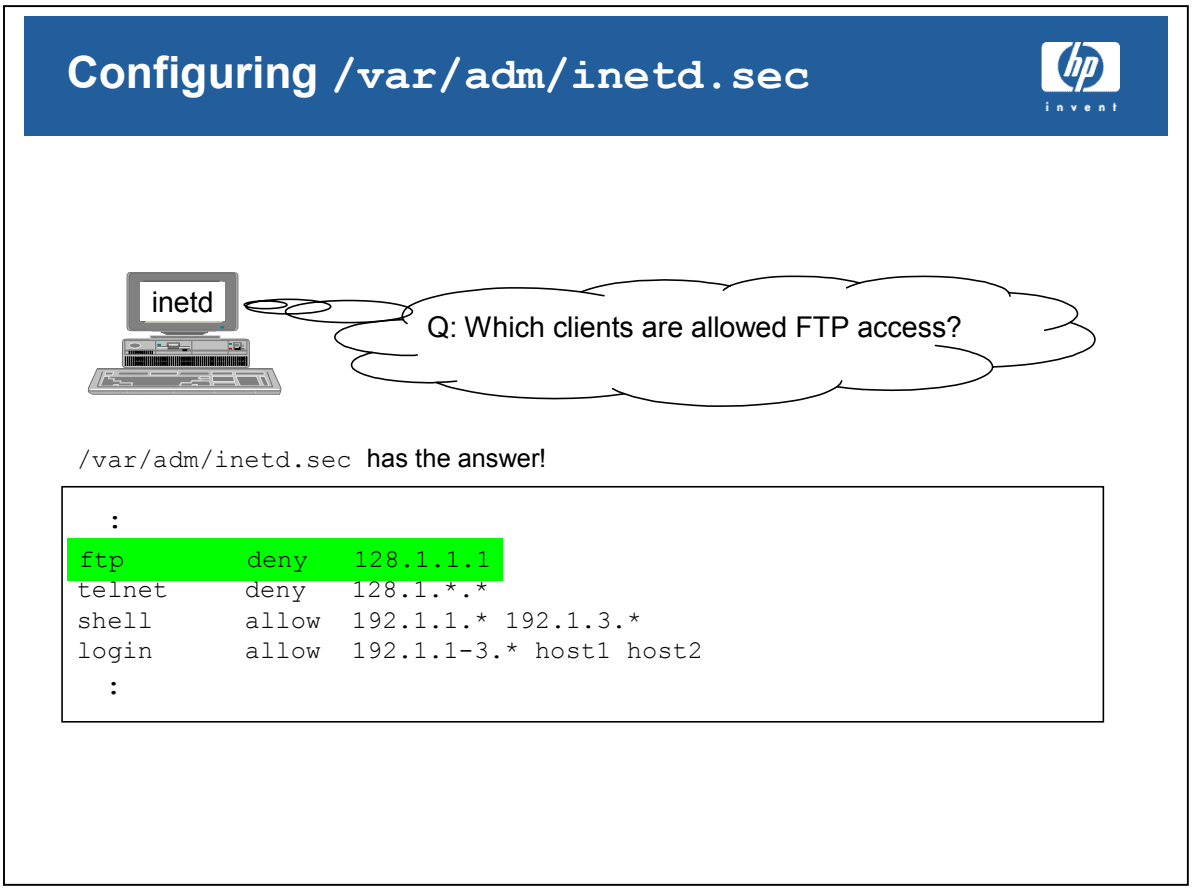
Figure 4

If additional clients request **telnet** service, the server's **inetd** daemon simply starts additional **telnetd** processes on port 23 as necessary.

NOTE:	Use **netstat -a** to see which ports are active.

© 2003 Hewlett-Packard Development Company, L.P.

13–7. SLIDE: Configuring `/var/adm/inetd.sec`



Student Notes

If you want to allow selected clients access to one or more Internet services, configure **`/var/adm/inetd.sec`**.

Each line in the file defines which clients may access a particular service managed by **`inetd`**. The slide examples are explained below:

- The **`inetd`** daemon denies **`ftp`** service to the host at 128.1.1.1. All other hosts, however, can **`ftp`** to the server.
- No hosts on the 128.1 network can **`telnet`** to the server.
- Only clients on the 192.1.1 or 192.1.3 networks can **`remsh`** to the server.
- Any host on the 192.1.1, 192.1.2, or 192.1.3 networks can **`rlogin`** to the server. The host names **`host1`** and **`host2`**, will also have **`rlogin`** access.

If **`inetd.sec`** does not exist, all configured services will be available to all clients. If the file exists but does not have an entry for one or more **`inetd`** services, the unlisted services will be available to all clients.

© 2003 Hewlett-Packard Development Company, L.P.

The formal syntax for the **inetd.sec** file is described below:

service_name	The name of a valid service in **/etc/services**.
allow|deny	Determines if the list of remote hosts in the next field is allowed or denied access to a service. The default is to allow access.
host_specifiers	The IP address, network names, or host name that should be allowed or denied access. A wild card character (*) and a range character (-) are allowed. These characters can be present in any fields of the address.

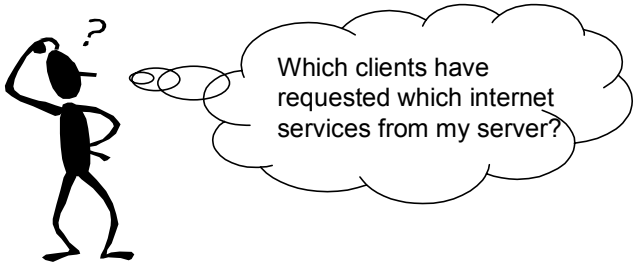
This file has to be owned by root. Its permissions are **r--r--r--.**

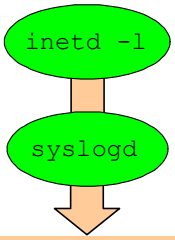
NOTE:	You have to use the official service name as specified in the **/etc/services** file. The service for **rlogin** is called login. The **shell** service is needed for **rcp** and **remsh**.

© 2003 Hewlett-Packard Development Company, L.P.

13–8. SLIDE: Configuring `inetd` Logging

Configuring `inetd` Logging





`/var/adm/syslog/syslog.log` has the answer!

```
Sep 5 15:51:10 host1 inetd[2234]: telnet/tcp: Connection from host1
Sep 5 15:51:27 host2 inetd[2251]: login/tcp: Connection from host2
```

`/etc/rc.config.d/netdaemons`

```
export INETD_ARGS=□-l□  # Enable inetd logging at every boot by
                        # setting the INETD_ARGS variable here!
```

Student Notes

The **`inetd -l`** command toggles **`inetd`** logging. If connection logging is enabled, the logging information is reported to the system logger (**`/usr/sbin/syslogd`**) and its log file **`/var/adm/syslog/syslog.log`**. If you activate logging, **`inetd`** will log attempted connections to the services. It will also log those connection attempts that fail the security check. This can be useful when trying to determine if someone is trying to break into your system. An example of the contents of the **`syslog`** file is shown below:

```
Jun 4 13:03:38 host2 inetd[994]: Reading configuration
Jun 4 13:03:38 host2 inetd[994]: ftp/tcp: Added service, server /usr/lbin/ftpd
Jun 4 13:03:38 host2 inetd[994]: telnet/tcp: Added service, server
/usr/lbin/telnetd
  :
Jun 5 16:20:49 host2 inetd[994]: Connection logging enabled
  :
Jun 5 16:21:00 host2 inetd[1383]: login/tcp: Connection from host (192.6.1.72)
at Sun Jun 5
16:21:00 1994
Jun 5 16:21:25 host2 inetd[1398]: ftp/tcp: Connection from host1 (192.6.1.72)
at Sun Jun 5
16:21:25 1994
```


© 2003 Hewlett-Packard Development Company, L.P.

To enable **inetd** logging at system start up, configure the appropriate variable in the **/etc/rc.config.d/netdaemons** file and restart the daemon:

```
# vi /etc/rc.config.d/netdaemons
  export INETD_ARGS=-l
# /sbin/init.d/inetd stop
# /sbin/init.d/inetd start
```

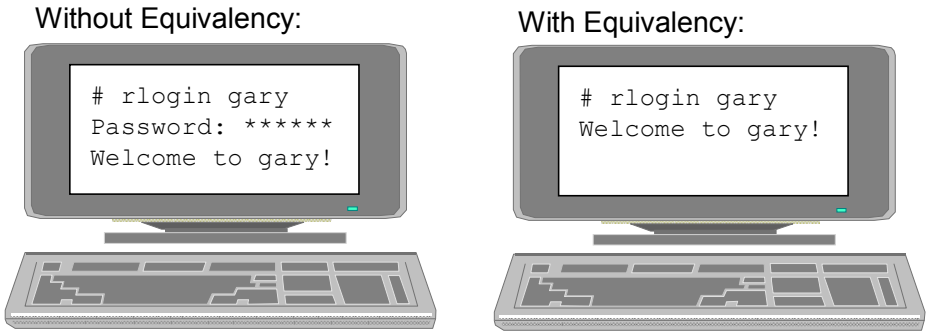
Note that **inetd** logging records host names that have requested internet services, but does not record the usernames that requested those services. The **/var/adm/wtmp** and **/var/adm/btmp** files log successful and unsuccessful login attempts. Use the following commands to view these files:

```
# last      (to view successful logins)
# lastb     (to view unsuccessful logins)
```


© 2003 Hewlett-Packard Development Company, L.P.

13–9. SLIDE: System and User Equivalency

System and User Equivalency



System and user equivalency:

- allows some or all users password-free access to a host
- only apply to Berkeley services (`rlogin`, `remsh`, `rcp`)
- configured via: `/etc/hosts.equiv` and `~/.rhosts`

Student Notes

System and user equivalency allows selected users to bypass password security when using **`rlogin`**, **`remsh`**, and **`rcp`** to access hosts across the network.

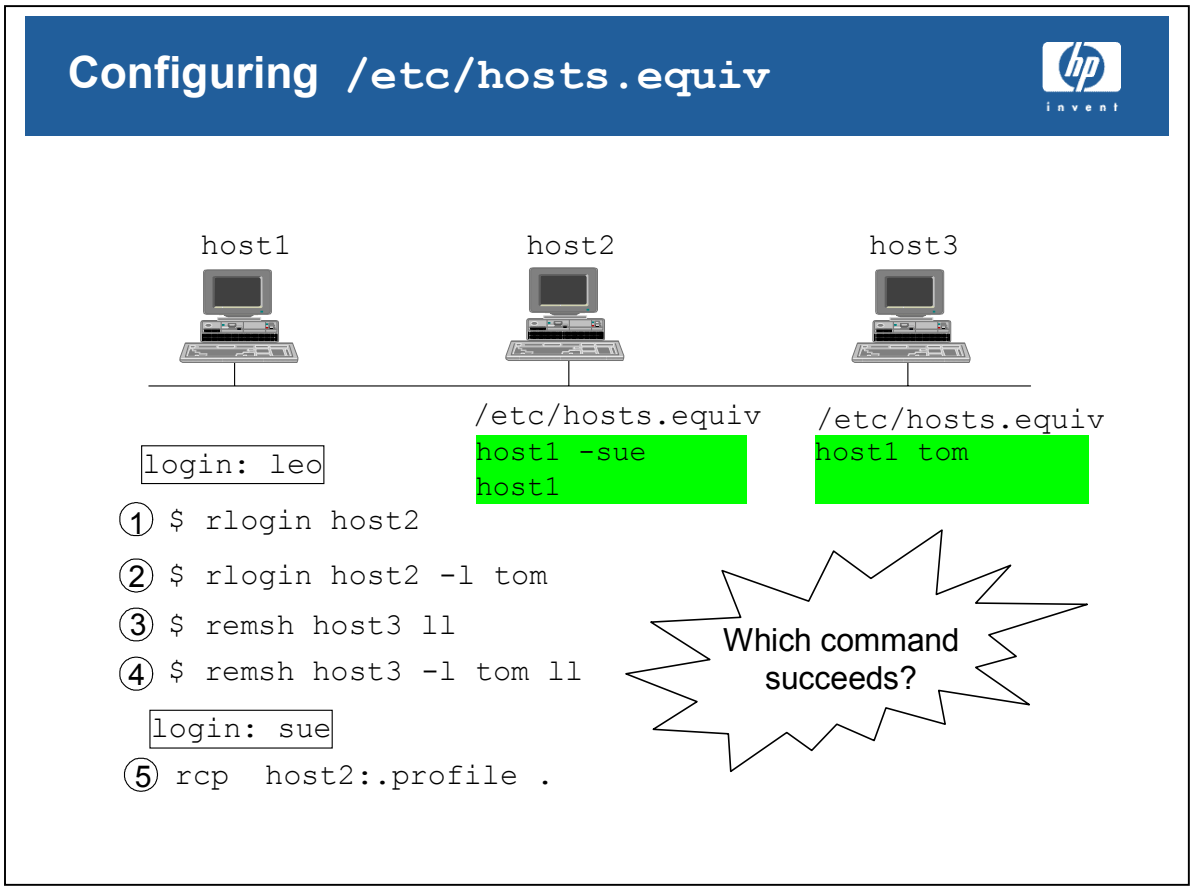
System equivalency is configured via the **`/etc/hosts.equiv`** file, and user equivalency is configured via **`~/.rhosts`**. Both of these files will be discussed in detail in the slides that follow.

Although these files allow your users conveniently and transparently to access their accounts on multiple systems, they create a significant security risk. Be sure the permissions on both files are set appropriately:

```
r--r--r--   /etc/hosts.equiv
rw-------   ~/.rhosts
```


© 2003 Hewlett-Packard Development Company, L.P.

13–10. SLIDE: Configuring `/etc/hosts.equiv`



Student Notes

The **`/etc/hosts.equiv`** file associates remote hosts with a user's host. This association identifies equivalent hosts that are frequently accessed by the same users. If a remote host is listed in **`hosts.equiv`**, and the remote user's login name matches a login name on the local host, the user is not prompted for a password. *This equivalency does not apply to superusers*. If you are logged in as root and you attempt to access another system, **`/etc/hosts.equiv`** is bypassed.

Typically, the system administrator creates the **`/etc/hosts.equiv`** file if she or he wishes to use this feature.

`/etc/hosts.equiv` works only with the Berkeley Services **`remsh`**, **`rcp`**, and **`rlogin`**

NOTE:	When you list a system in **`hosts.equiv`**, all users on that system with the same user name as on your system, have access to your system except the root user. Root user equivalency can be set up through **`.rhosts`**.

© 2003 Hewlett-Packard Development Company, L.P.

Entries in `/etc/hosts.equiv`

A host name or user name can match the corresponding field in an entry in **`hosts.equiv`** in many ways. Several of these are

Literal match — A host in **`hosts.equiv`** may literally match the host name (not an alias) of the remote host. A user name in **`hosts.equiv`** may literally match the remote user name. If there is no user name in the **`hosts.equiv`** entry, the remote user name must literally match the local user name.

-name — If the host name in **`hosts.equiv`** is of this form, and if name literally matches the remote host name or ***`name`*** with the local domain name appended matches the remote host name, then access is denied regardless of the user name. If the user name in **`hosts.equiv`** is of this form, and ***`name`*** literally matches the remote user name, access is denied. Even if access is denied in this way by **`hosts.equiv`**, access can still be allowed by **`.rhosts`**.

+ Any remote host name matches the **+** host name in **`hosts.equiv`**. Any remote user matches the **+** user name.

See **`hosts.equiv(4)`** for more information.

Examples

1. **`$ rlogin host2`**

   `leo` wants to log in to system **`host2`** as user **`leo`**. Equivalency is configured. No password is required.

2. **`$ rlogin host2 -l tom`**

   `leo` has to enter the password because equivalency between different users is not possible with **`/etc/hosts.equiv`**.

3. **`remsh host3 ll`**

   `leo` wants to access system **`host3`** as user **`leo`**. This will fail because there is only equivalency configured for user **`tom`** from **`host1`**.

4. **`remsh host3 -l tom ll`**

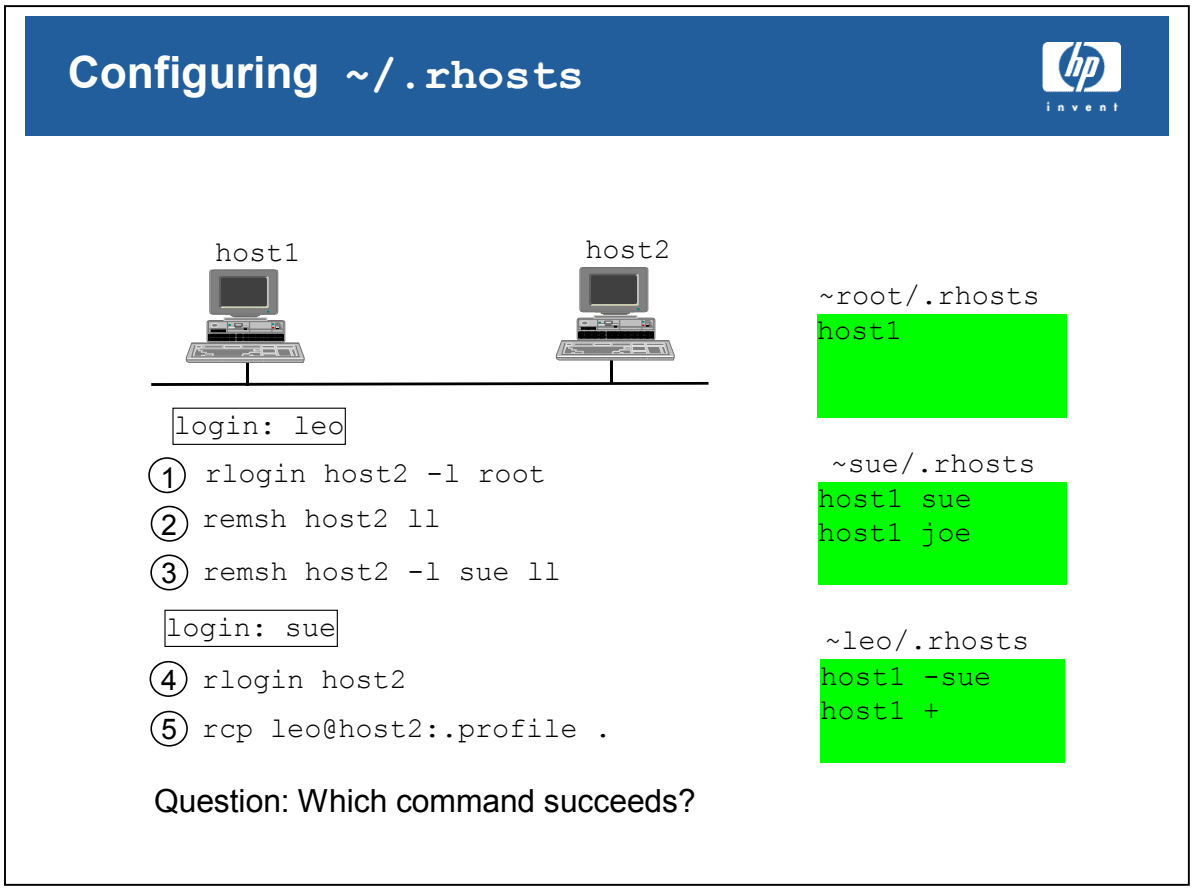
   `leo` wants to access system **`host3`** as user **`tom`**. This will fail because there is only equivalency configured for user **`tom`** from **`host1`**.

5. **`rcp host2:.profile .`**

   `sue` from **`host1`** wants to access **`sue`** on system **`host2`**. **`rcp`** fails because **`sue`** is the only user from system **`host1`** who is excluded.

© 2003 Hewlett-Packard Development Company, L.P.

13–11. SLIDE: Configuring ~/.rhosts



Student Notes

$HOME/.rhosts can be created and configured by any user to specify remote login names that are equivalent to the local user's login name. **$HOME/.rhosts** must be owned by the local user.

The local host allows a remote user with a login listed in the local **$HOME/.rhosts** file to log into the local user's account without specifying a password. The remote user can also copy files or execute commands on the local user's system.

The **.rhosts** file works only with the Berkeley Services **remsh**, **rcp**, and **rlogin**.

The characters **+** and **-** can also be used. Look at the examples shown on the slide.

NOTE: **.rhosts** can be used to allow service to a particular user whose system has not been granted access in **/etc/hosts.equiv**. You must create **.rhosts** for the home directory of the superuser account if you wish to use equivalent login names for root.

© 2003 Hewlett-Packard Development Company, L.P.

Examples

1. **rlogin host2 -l root**

   A password is required. Root's **/.rhosts** is only configured for the user **root** from system **host1**.

2. **remsh host2 ll**

   leo wants to access user **leo** on system **host2**. This is successful because **/home/leo/.rhosts** on system **host2** has an entry for all users from system **host1** except user **sue**.

3. **remsh host2 -l sue ll**

   This fails because there is no entry for user **leo** from system **host1** in **sue**'s file.

4. **rlogin host2**

   Now **sue** wants to log in to her account on system **host2**. There is no password required because of the entry **host1 sue** is in **/home/sue/.rhosts**.

5. **rcp leo@host2:.profile .**

   This fails. No user equivalency is configured for **sue** in the **/home/leo/.rhosts** file. She is the only user from system **host1** who is excluded.

Disabling Users' .rhosts Files

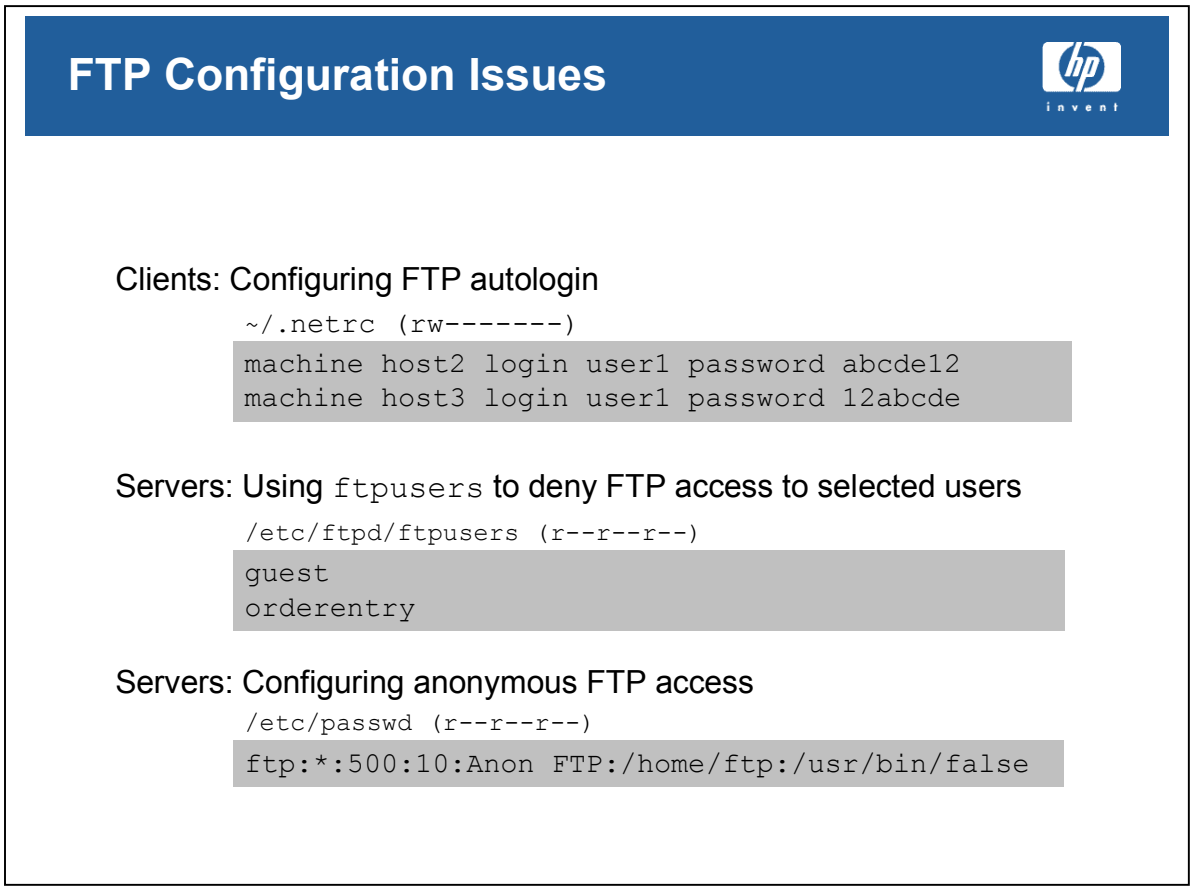
Users may not realize the security risk of an improperly configured **.rhosts** file. You can prevent the Berkeley services from consulting users' **.rhosts** files by adding a **-l** to the "shell" and "login" lines in **inetd.conf**:

```
# vi /etc/inetd.conf
     login   stream tcp nowait root /usr/lbin/rlogind rlogind -l
     shell   stream tcp nowait root /usr/lbin/remshd remshd -l
# inetd -c
```

Note that this does not affect root's **.rhosts** file. **/etc/hosts.equiv** will still be consulted.

© 2003 Hewlett-Packard Development Company, L.P.

13–12. SLIDE: FTP Configuration Issues



Student Notes

There are three different security issues related to the configuration of **FTP**.

Clients Configuring FTP Autologin

Creating a **.netrc** file allows a user to **ftp** to other hosts without manually entering a username or password. Instead, **ftp** simply looks in the user's **.netrc** to determine the username and password.

Note that **.netrc** poses a possible security risk since passwords are stored in cleartext. Make sure the **.netrc** permissions are set to:

```
rw-------
```

The login will fail if the permissions on the file are not set properly.

© 2003 Hewlett-Packard Development Company, L.P.

Servers Using `/etc/ftpd/ftpusers` to Deny `FTP` Access to Selected Users

The **`ftpd`** daemon will reject logins to local accounts that are named in **`/etc/ftpd/ftpusers`**. Each account name must appear on a line by itself. The line cannot contain any white space.

The **`ftpd`** daemon does not check the startup program field in **`/etc/passwd`**, so accounts that have a restricted shell as the startup program should be listed in **`/etc/ftpd/ftpusers`**. Other users who should not have **`ftp`** access may be included in the file as well.

Servers Configuring Anonymous `ftp` Access

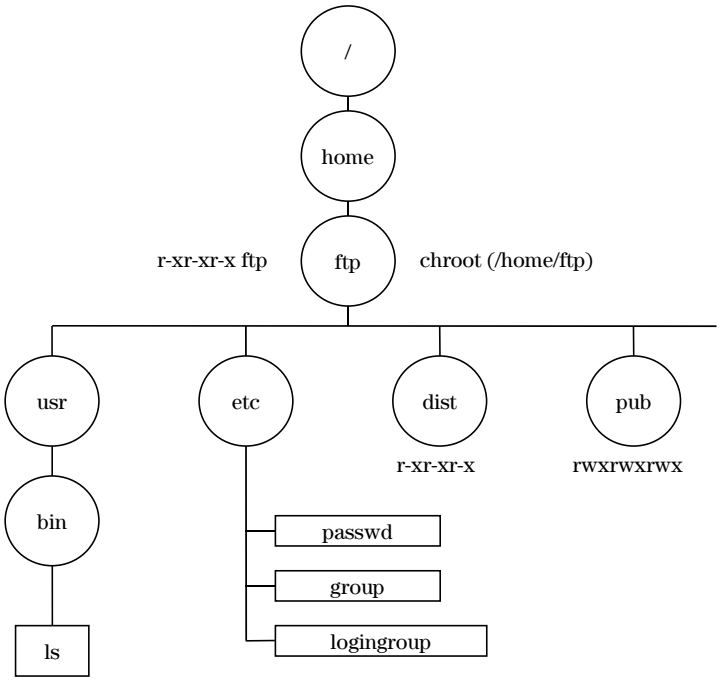


Figure 5

`Anonymous ftp` is a secure public user account. If this has been set up, users can access the **`anonymous ftp`** account with the user name **`anonymous`** or **`ftp`** and any non-null password (by convention, the client email address). **`ftpd`** does a **`chroot()`** to the home directory of user **`ftp`**, thus limiting anonymous **`ftp`** users' access to the system. The anonymous **`ftp`** account must be present in the password file (user **`ftp`**). The password field should be an asterisk (**`*`**), the group membership should be **`guest`**, and the login shell should be **`/usr/bin/false`**. For example, (assuming the guest group ID is 10)

```
ftp:*:500:10:Anonymous ftp user:/home/ftp:/usr/bin/false
```


© 2003 Hewlett-Packard Development Company, L.P.

Since **ftpd** does a **chroot()** to **/home/ftp**, it must have the following subdirectories and files:

~ftp/usr/bin This directory must be owned by root and must have the permissions 555 (not writable). It should contain a copy of **/usr/bin/ls**. This is needed to support directory listing by **ftpd**. The command should have the permissions 111 (executable only). If the **ftp** account is on the same file system as **/usr/bin**, **~ftp/usr/bin/ls** can be a hard link, but it cannot be a symbolic link because of the **chroot()**.

~ftp/etc This directory must be owned by root and must have the permissions 555 (not writable). It should contain versions of the files **/etc/passwd**, **/etc/group**, and **/etc/logingroup**. These files must be owned by root and must have the permissions **444** (readable only). These files are needed to map **user** and **group ids** to names when using the built-in **ls** command of **ftp**, and to support (optional) sublogins of **anonymous ftp**.

~ftp/pub This directory (optional) is used by anonymous **ftp** users to deposit files on the system. It should be owned by user **ftp** and should have the permissions 1777 (readable and writable by all). If this directory is created, disk quotas should be used to prevent anonymous users from filling the file system.

~ftp/dist This directory (optional) is used to make files available to anonymous **ftp** users. It should be owned by user **ftp** and must have the permissions 555. Any files to be distributed should have the permissions 444 (readable only) so they cannot be modified or removed by anonymous **ftp** users.

NOTE: The directory **~ftp/pub** for depositing files must have the permissions 1777. To prevent anonymous **ftp** users from filling the file system you should use disk quotas.

If you only want to make files available, you do not need the directory **~ftp/pub**.

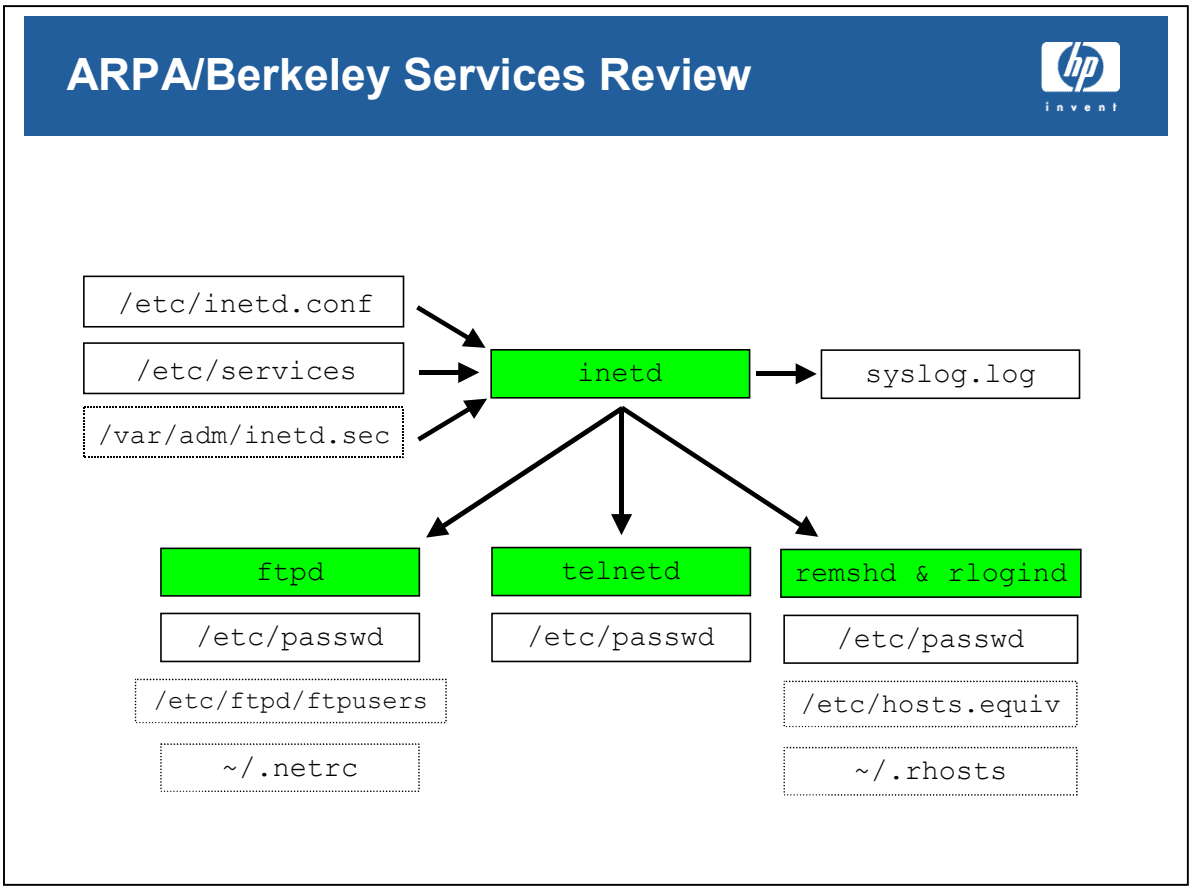
When adding or removing users with SAM, the files in **/home/ftp/etc** are not customized.

Anonymous ftp can be configured with SAM:

```
sam
  |
  V
Networking and Communications ->
  |
  V
Network Services
  |
  V
Anonymous ftp
```


© 2003 Hewlett-Packard Development Company, L.P.

13–13. SLIDE: ARPA/Berkeley Services Review



Student Notes

This slide reviews the important executables and configuration files that control access to the Internet services. An explanation of the ARPA/Berkeley service configuration files follows below:

/etc/inetd.conf	Determines which services **inetd** should and should not provide.
/etc/services	Associates service names with well-known port numbers.
/var/adm/inetd.sec	Determines which clients have access to which **inetd** services. (Optional)
/var/adm/syslog/syslog.log	Records which clients have requested which **inetd** services, and when (if logging is enabled).
/etc/passwd	Defines valid accounts and passwords.

© 2003 Hewlett-Packard Development Company, L.P.

`/etc/ftpusers`	Defines which usernames are not valid for **`ftp`** logins (optional)
`~/.netrc`	Enables **`ftp`** autologon functionality (optional)
`/etc/hosts.equiv`	Configures host equivalency (optional)
`~/.rhosts`	Configures user equivalency (optional)

© 2003 Hewlett-Packard Development Company, L.P.

13–14. LAB: Configuring and Securing ARPA/Berkeley Services

Directions

This lab offers an opportunity to configure, use, and troubleshoot the ARPA/Berkeley service configuration on your machine. For a portion of the lab, you will need to work with a partner. Choose a partner, and decide which machine will be the internet service "server" during the experiments that follow, and which will be the "client".

Note that the "server" and "client" roles assigned in this lab are relatively arbitrary. Most HP-UX machines are configured to provide both client and server functionality.

Server's host name: _____

Client's host name: _____

Part 1: Basic ARPA/Berkeley Service Configuration

1. **(server and client)**
   The "InternetSrvcs" product must be installed on every machine that wishes to use or provide ARPA/Berkeley services. Check to ensure that this product is installed on your system.

2. **(server)**
   The server's **`inetd`** daemon must be running in order for clients to have access to any of the internet services. Use **`ps -e`** to check to ensure that the **`inetd`** daemon is running on your server.

3. **(server and client)**
   Which script starts **`inetd`** during the boot process? At which run level does **`inetd`** start?

© 2003 Hewlett-Packard Development Company, L.P.

4. **(server)**
   Look at **`/etc/inetd.conf`** and **`/etc/services`** to determine which internet services are configured on your server, then complete the table below:

Service	Enabled?	Port#
telnet		
ftp		
login		
tftp		
bootps		

5. Do you currently have server processes running for these services? Explain.

6. **(server)**
   Ensure that the services in **`inetd.conf`** that appear to be enabled actually are enabled. Use **`netstat -a`** to check the status of each of the enabled services and ports you listed in the table above.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Securing the Internet Services

1. **(server)**
   The **`inetd.conf`** file allows you to enable or disable an internet service for *all* clients. If, however, you wish to allow/prevent *specific* client(s) access to a service, you must use the **`/var/adm/inetd.sec`** file.

   Configure your **`/var/adm/inetd.sec`** file such that only the hosts in your row (including your partner) have **`telnet`** access. Add another line to ensure that all your classmates except your partner can **`ftp`** to your machine.

2. **(client)**
   See if your server's configurations so far have succeeded. What messages do you see when you attempt to **`telnet`** or **`ftp`** to the server?

3. **(server)**
   What do you have to do to enable **`inetd`** logging? Make it so.

4. **(client)**
   See if the logging feature works. From the client, **`telnet`** to the server, do an **`ls`**, then immediately exit. Then attempt to **`ftp`** to the server (this should fail). Move on to the next question to see what was recorded in the **`inetd`** log.

5. **(server)**
   How much detail is recorded in the **`inetd`** log? On the server, do a **`more`** on the file where ARPA/Berkeley service requests are logged.

© 2003 Hewlett-Packard Development Company, L.P.

- Does **inetd** log the name of the service requested?
- Does **inetd** log the host name of the requesting client?
- Does **inetd** log the username of the user making **telnet** requests?
- Does **inetd** log the commands executed during the **telnet** session?
- Does **inetd** log deny requests for Internet service?

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Experimenting with ARPA/Berkeley Service Connections

The goal of this part of the lab is to determine what happens when a client process connects to a server providing ARPA/Berkeley services. More specifically, we will be experimenting with the **telnet** service.

1. **(client and server)**
   First, check to see which daemons and processes are already running on the server and client:

   ```
   client# ps -e | grep telnet
   server# ps -e | grep telnet
   ```

2. **(client and server)**
   Establish a **telnet** session from the client to the server, and look at the process table to determine which processes were started as a result.

   ```
   client# telnet server
   client# ps -e | grep telnet
   server# ps -e | grep telnet
   ```

   Which **telnet** related processes are running on the client now? Which **telnet** related processes are running on the server now?

3. **(client and server)**
   Can multiple clients **telnet** to the server simultaneously? Try it. On the client side, open another window and initiate another **telnet** connection to the server. Check to see which **telnet** related processes are running on the server and client:

   ```
   client# ps -e | grep telnet
   server# ps -e | grep telnet
   ```

   How many **telnetd** server processes are running on the server? How many **telnet** processes are running on the client? Explain.

© 2003 Hewlett-Packard Development Company, L.P.

4. **(client and server)**
   Take a look at the ports that are being used by your **`telnet`** processes:

   ```
   client# netstat -a | grep telnet
   server# netstat -a | grep telnet
   ```

   How many **`telnet`** connections are ESTABLISHED? What process do you suppose is monitoring the port in the LISTEN state? Do the client side **`telnet`** processes share a port or use different ports? Which well-known port numbers are the **`telnetd`** daemons on the server sharing?

5. **(client)**
   Close your **`telnet`** connections to the server.

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Experimenting with ARPA/Berkeley Services

1. **(client)**
   What happens if the server's **inetd** daemon is down when a client attempts to connect? Try it, then explain the result.

```
server# inetd -k              # kill the server's inetd
client# telnet server         # can the client still connect?
server# inetd                 # restart the server's inetd
```

2. **(client and server)**
   What happens if the server's **inetd** daemon goes down AFTER a session has been established -- does the existing connection remain, or are all client connections immediately terminated? Try it, then explain the result.

```
client# telnet server         # establish a connection to the
                              server
server# inetd -k              # kill the server's inetd.
server# ps -e | grep telnetd  # does the telnet daemon remain?
```

3. **(client and server)**
   What happens if the server's **telnetd** server process is killed while a client is connected? Try it.

```
server# ps -e | grep telnetd  # find the server process's PID
server# kill _____            # kill telnetd's PID
```

   Does the client **telnet** process exist after the server's **telnetd** daemon is killed?

   Restart **inetd** on the server before proceeding to the next question.

```
# inetd
```


© 2003 Hewlett-Packard Development Company, L.P.

4. **(client)**
   Must the client be running **inetd** in order to establish connections to a server? Try it, and explain the result.

```
client# inetd -k              # kill the client's inetd
client# telnet server         # can the client still telnet out?
client# inetd                 # restart the client's inetd
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Experimenting with Host and User Equivalency

1. **(server)**
   Configure host equivalency for all the hosts in your row, including your client.

2. **(client)**
   While logged in as root, use **`rlogin`** to log into the server. What happens? Why?
   Exit out of your **`rlogin`** session before proceeding to the next question.

3. **(client)**
   Use the **`su`** command to switch your user ID to user1. Then try **`rlogin`** again. What happens? Why?

4. **(server)**
   What can you do on the server to enable root on the clients password free access to your machine? Make it so.

5. **(client)**
   Terminate the **`rlogin`** and **`su`** sessions you stated previously. Ensure that you are back to the "root" userid. Then see if you can **`rlogin`** to the server without a password.

© 2003 Hewlett-Packard Development Company, L.P.

6. **(server)**
   Remove **`/etc/hosts.equiv`** and **`~root/.rhosts`**.

© 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Troubleshooting Problems with the Internet Services

In the exercise that follows, you will work with your partner to corrupt, then fix the internet service configuration on the server machine that you chose at the beginning of this lab.

The list below suggests several different ways to corrupt the internet service configuration on your "server" machine. Take turns being the "corrupter" and the "troubleshooter."

The "corrupter" should perform any one of the corruption techniques from the list below on the "server" machine. It is the duty of the "troubleshooter," then to do whatever is necessary on the server to enable the client to successfully **telnet** to the server.

Try the exercise several times, alternating roles as "corrupter" and "troubleshooter."

Before starting the exercise, shutdown CDE:

```
/sbin/init.d/dtlogin.rc stop
```

Eight Ways to Corrupt an Internet Service Server

1. Kill the **inetd** daemon with **inetd -k**.
2. Comment out the **telnet** line in **/etc/inetd.conf** and restart **inetd**.
3. Comment out the **telnet** line in **/etc/services** and restart **inetd**.
4. Take down the server's LAN card with **ifconfig lan0 down**.
5. Change the server's IP address with **ifconfig lan0 254.254.254.254**.
6. Detach the LAN cable on the server.
7. Change the client's network entry in the server's routing table.
8. Deny the client **telnet** access via **/var/adm/inetd.sec**.

Part 7: Cleanup

Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

13–15. REVIEW QUESTIONS: Configuring and Securing ARPA/Berkeley Services

Directions

Answer the following questions.

1. What is the difference between a daemon and a server process?

2. List some Internet Services daemons and server processes.

3. What does **`inetd`** do? What is the advantage in its functioning?

4. What is the name of the **`inetd`** configuration file?

5. What command do you use after modifying the configuration file?

6. What is a port? What file associates port numbers with a service name?

© 2003 Hewlett-Packard Development Company, L.P.

7. List at least four security features of the Internet Services.

8. Which server processes use the **`/etc/hosts.equiv`** and **`$HOME/.rhosts`** files?

9. Are the **`/etc/hosts.equiv`** and **`$HOME/.rhosts`** files optional for using the Berkeley Services? Explain your answer.

10. What is the name and what are the features of the security file that **`ftpd`** uses?

11. What is an **`anonymous ftp`**?

12. What is the security feature of **`/var/adm/inetd.sec`**?

© 2003 Hewlett-Packard Development Company, L.P.

13. What is wrong in the following **inetd.sec** example?

```
rlogin allow 192.6.1
```

14. If **inetd** logging is enabled, which file contains the logging output?

© 2003 Hewlett-Packard Development Company, L.P.

Module 14 — Configuring a BOOTP/TFTP Server

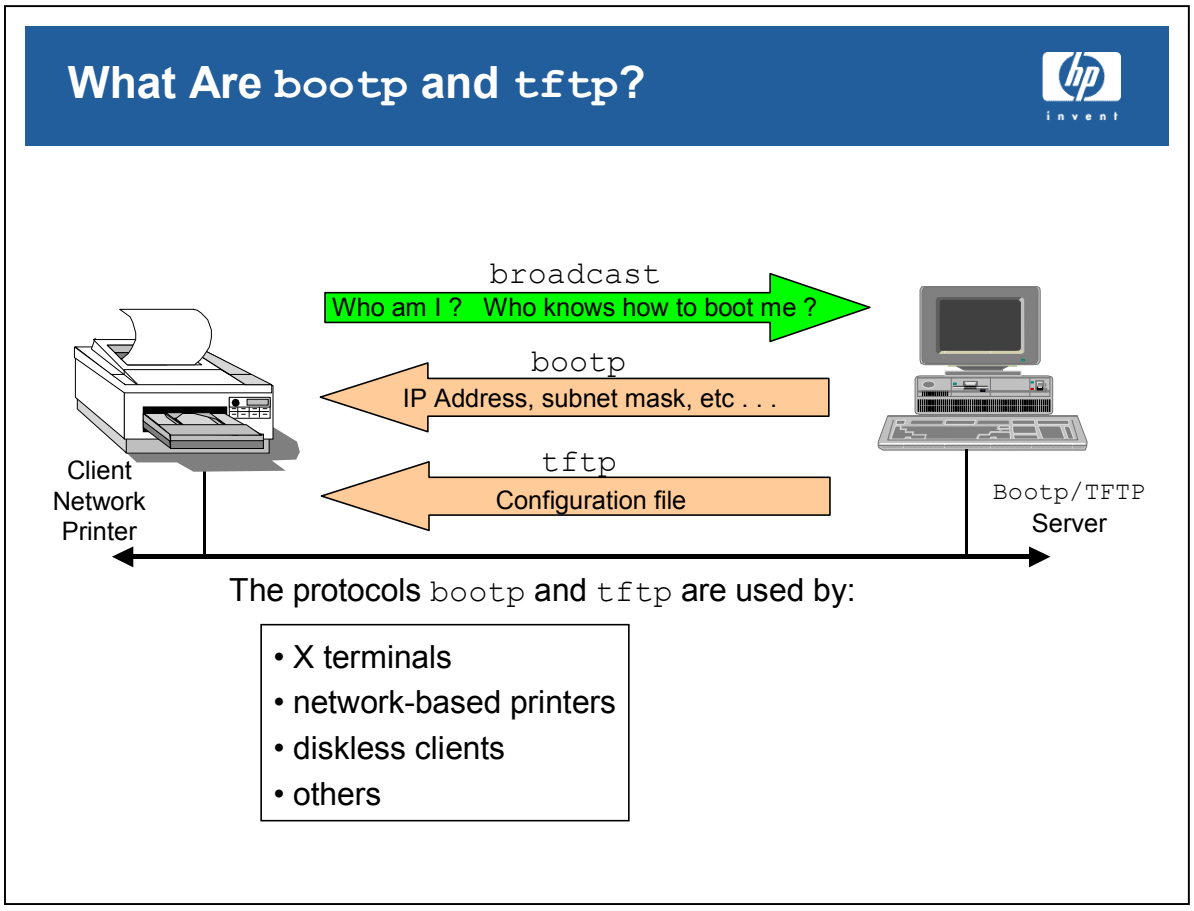
Objectives

Upon completion of this module, you will be able to do the following:

- Describe the purpose of **`bootp`** and **`tftp`**.
- Configure **`inetd`** to provide **`bootp`** and **`tftp`** services.
- Describe the purpose and contents of the **`bootptab`** file.
- Describe the purpose of a network-based printer.
- Configure a **`bootptab`** entry for a network printer using **`hppi`**.
- Configure a **`bootptab`** entry for an X terminal using **`xtadm`**.

© 2003 Hewlett-Packard Development Company, L.P.

14–1. SLIDE: What Are `bootp` and `tftp`?



Student Notes

The Bootstrap Protocol (**`bootp`**) allows certain network devices, like X terminals, network printers, and diskless client workstations, to discover their network configuration information (such as IP address and subnet mask) and boot information from another system on the network automatically.

The Trivial File Transfer Protocol (**`tftp`**) is a simple protocol used to read and write files to or from a remote system. This protocol is used to transfer configuration information for clients booting using the **`bootp`** protocol.

Together, **`tftp`** and **`bootp`** allow a system to provide boot information for client systems that support **`bootp`**. These protocols are implemented on top of the User Datagram Protocol (UDP), so they can be used across networks that support UDP.

© 2003 Hewlett-Packard Development Company, L.P.

14–2. SLIDE: Enabling `bootp` and `tftp` Services

Enabling `bootp` and `tftp` Services

1. Ensure services are defined in `/etc/services`:

```
# cat /etc/services
bootps          67/udp
tftp            69/udp
```

2. Ensure services are defined in `/etc/inetd.conf`:

```
# cat /etc/inetd.conf
tftp    dgram  udp  wait  root  /usr/lbin/tftp     tftp
bootps dgram  udp  wait  root  /usr/lbin/bootpd  bootpd
```

3. Ensure `tftp` account is defined in `/etc/passwd`:

```
# cat /etc/passwd
tftp:*:510:1:Trivial FTP User:/home/tftpdir:/usr/bin/false
```

4. Reconfigure `inetd` daemon (if necessary):

```
# inetd -c
```

Student Notes

In order for the BOOTP and TFTP protocols to boot clients and download network-related parameters, a number of configurations must be made to key configuration files. These configurations allow the BOOTP and TFTP daemons, **`/usr/lbin/bootpd`** and **`/usr/lbin/tftpd`** on an HP-UX system, to properly service the requests of clients.

The following is a list of three files that must be updated for **`bootp`** and **`tftp`** to work:

- A **`tftp`** user entry is needed for the TFTP service in **`/etc/passwd`**.

```
# more /etc/passwd
+-----------------------------------------------
| ...
|tftp:*:510:1:Trivial FTP user:/home/tftpdir:/usr/bin/false
|
```

The **`tftpd`** daemon uses the system call **`chroot()`** to change its root directory to be the same as the home directory of the pseudo-user **`tftp`**. This restricts access by **`tftp`** clients to only those files found below the **`tftp`** home directory.

© 2003 Hewlett-Packard Development Company, L.P.

- In order for the system to know about the BOOTP and TFTP protocols, they must be defined in the **/etc/services** file. Notice for the BOOTP entry, there are both client and server definitions:

```
# more /etc/services
+-----------------------------------------------
|bootps  67/udp # Bootstrap Protocol Server
|bootpc  68/udp # Bootstrap Protocol Client
|tftp    69/udp # Trivial File Transfer Protocol
```

- Usually, both daemons (**bootpd** and **tftpd**) are started by the Internet daemon (**inetd**) and must be registered in the **/etc/inetd.conf** file:

```
# more/etc/inetd.conf
+-----------------------------------------------------
|ftp     stream tcp nowait root /usr/lbin/ftpd ftpd
|bootps  dgram  udp wait   root /usr/lbin/bootpd bootpd
```


© 2003 Hewlett-Packard Development Company, L.P.

14–3. SLIDE: Configuring /etc/bootptab

Configuring /etc/bootptab

Sample entries in the /etc/bootptab file:

```
# cat /etc/bootptab
myprinter:\
              hn:\
              ht=ether:\
              ha=080009a752c3:\
              ip=156.153.82.239:\
              T144="myprinter.cfg":\
              vm=rfc1048

myxterm:\
              ba:\
              hn:\
              ht=ether:\
              ha=0800091ef227:\
              ip=156.153.198.155:\
              bf=/opt/hpxt/enware/xthome/:\
              T144="myxterm.cfg":\
              vm=rfc1048
```

Student Notes

The final configuration is to add an entry for the client to the server's **/etc/bootptab** file so the server knows about the specific client. The BOOTP daemon (**/usr/lbin/bootpd**) compares **bootp** broadcast requests to entries in this file. If there is an entry for the client performing the broadcast, then the server responds.

The following are examples of **bootptab** entries for a network printer and an X station:
/etc/bootptab

```
/etc/bootptab

+----------------------------
|myprinter:\
|            hn:\
|            ht=ether:\
|            ha=080009a752c3:\
|            ip=192.6.1.11:\
|            gw=192.6.1.1:\
|            T144="hpnp/myprinter.cfg":\
|            vm=rfc1048:
```


© 2003 Hewlett-Packard Development Company, L.P.

```
|
|myxterm:\
|           ba:\
|           hn:\
|           ht=ether:\
|           ha=0800099856da:\
|           ip=192.6.1.108:\
|           bf=/opt/hpxt/enware/xthome:\
|           gw=192.6.1.58:\
|           sm=255.255.255.0:\
|           T144="xterm1.cfg":\
```

myprinter, **myxterm** — These are the network node names of the network printer and X station and must be configured also in a name service database such as **/etc/hosts**, NIS, or BIND.

ba — Specifies that **bootpd** should broadcast the boot reply to the client. As a Boolean tag, it causes **bootpd** to send the boot reply on the configured broadcast address of each network interface.

hn — Indicates that the client's node name should be sent in the boot reply.

ht — Specifies the network hardware type. Another supported value is **token-ring**.

ha — Specifies the link level address of the X station or network printer.

ip — Specifies the IP address of the BOOTP client (the X station or network printer).

bf — Specifies the location of the boot file (X server).

gw — Specifies the IP address of a gateway (router). A gateway must be specified if the X station should connect to a file or X session server that is not connected to the same network or subnet.

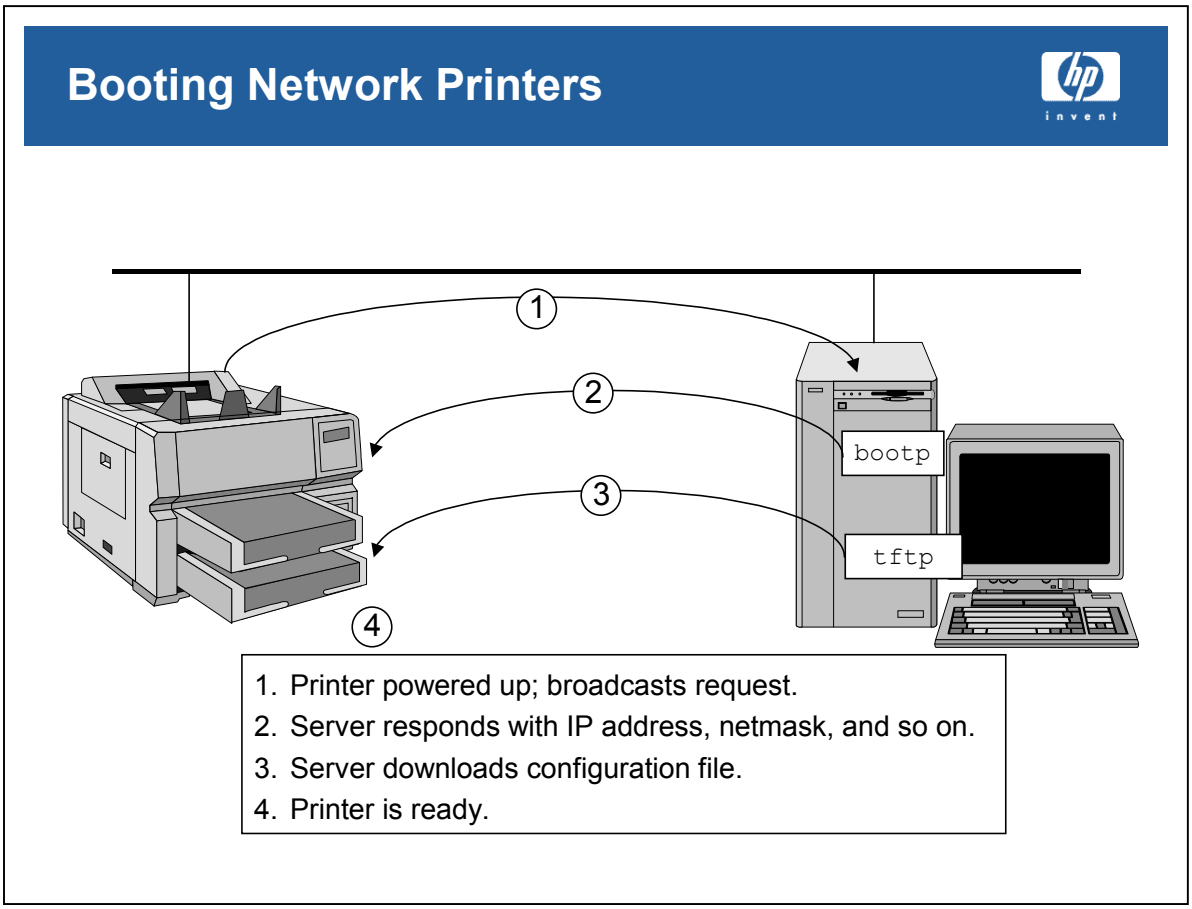
sm — Specifies the network subnet mask. This is only needed when using subnetting.

T144 — Specifies a configuration file name. The name must be enclosed in quotation marks (`" "`). This is optional. If it exists, it is transferred to the X station via TFTP or NFS. The location of this file is **/opt/enware/xthome/config**.

vm — The value rfc1048 forces an RFC1048 style reply (RFC = DARPA Internet request for comments).

© 2003 Hewlett-Packard Development Company, L.P.

14–4. SLIDE: Booting Network Printers



Student Notes

The above slide shows the sequence of steps for a network printer to boot from a printer server and communicate via the BOOTP and TFTP protocols.

1. The client (i.e. network printer) broadcasts a **`bootp`** request packet containing at least its hardware address.

2. The server answers with a **`bootp`** reply packet containing all of the information the client needs to know.

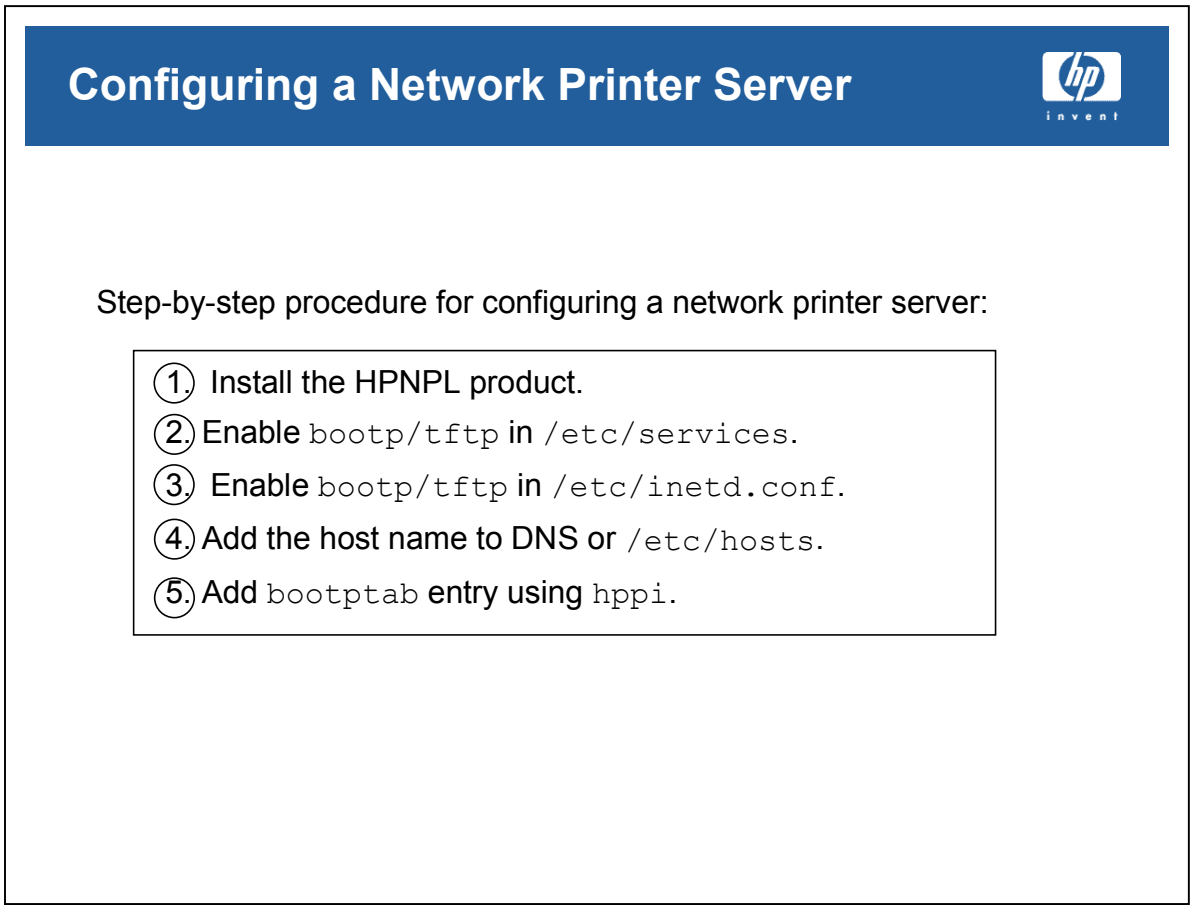
   - client's IP address
   - name of a boot file
   - other information, such as a subnet mask and the addresses of name servers or gateways

3. The client then uses **`tftp`** to transfer its configuration and other related files from the printer server.

4. Once the transfer has completed, the printer is ready to receive requests from nodes on the network.

© 2003 Hewlett-Packard Development Company, L.P.

14–5. SLIDE: Configuring a Network Printer Server



Student Notes

Before you begin, you will need to know the new printer's:

- MAC address
- IP address
- Hostname
- Subnet mask
- Default gateway address (optional)
- DNS name server address (optional)

The IP address, hostname, netmask, gateway, and DNS address all may be obtained from your network administrator. Print a test page on the printer to determine the printer's MAC address.

With this information in hand, you can begin configuring your printer!

1. Install the HPNPL product. HP recommends using a menu-based utility called **`hppi`** to configure BOOTP/TFTP service for network printers. **`hppi`** is part of the HPNPL (HP Network Printer Library) product, which is available from the **`http://www.hp.com`** website. Follow the instructions on the website to download and install the HPNPL software.

© 2003 Hewlett-Packard Development Company, L.P.

2. Enable **bootp**/**tftp** in **/etc/services**. Both services must be registered in the **/etc/services** file. Add entries, if necessary.

```
/etc/services
+---------------------------------------------
|bootps  67/udp  # Bootstrap Protocol Server
|bootpc  68/udp  # Bootstrap Protocol Client
|tftp    69/udp  # Trivial File Transfer Protocol
```

3. Enable bootpd and tftpd in /etc/inetd.conf.

```
# vi /etc/inetd.conf
+-----------------------------------------------------
|bootps dgram udp wait root /usr/lbin/bootpd bootpd
|tftpd  dgram udp wait root /usr/lbin/tftpd  tftpd
# inetd -c
```

   Note that some printers attempt to download a configuration file from the **bootp**/**tftp** server via TFTP. In order for this to succeed, you must add a **tftp** entry to **/etc/passwd** and create a **/home/tftpdir** home directory. Enabling TFTP via SAM will take care of these additional steps for you automatically:

```
# sam --> Networking and Communications --> Network Services
```

4. Add the printer's hostname and IP address to DNS or **/etc/hosts**.

5. Add a **bootptab** entry using **hppi**.

   hppi provides a convenient menu-based interface for adding, modifying, and removing network printer entries in the **/etc/bootptab** file.

```
# /opt/hpnpl/bin/hppi
   -> 2) JetDirect Configuration
   -> 1) Create printer configuration in BOOTP/TFTP database
```

   hppi will prompt you for the new printer's hardware (MAC) address, IP, hostname and other configuration information. After entering the requested information, hit **0** to configure the printer, then quit out of **hppi**. Check the **/etc/bootptab** file to verify that a new entry was added.

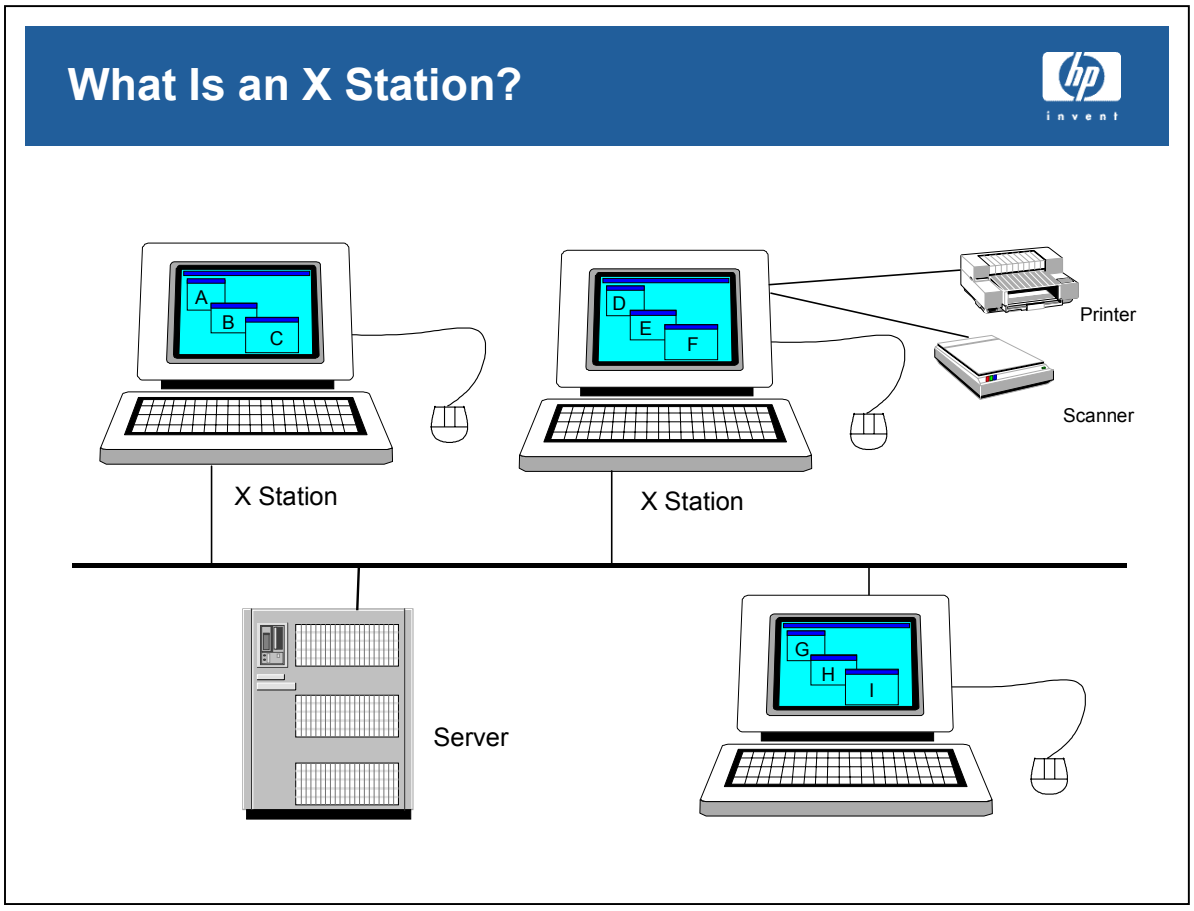
Finally, verify that the printer is configured to obtain network information via BOOTP, rather than from the printer front panel. See your printer owner's manual for details. Power-cycle your printer and print a test page to verify that your configuration succeeded!

If you want to print to the new printer from the boot server, the printer must be configured into the local print spooler. **hppi** can be used for this task, too:

```
# /opt/hpnpl/bin/hppi
  -> 1) Spooler Administration (super-user only)
  -> 1) Add printer to local spooler
```


© 2003 Hewlett-Packard Development Company, L.P.

14–6. SLIDE: What Is an X station?



Student Notes

An X station (X terminal) overcomes the limitations of an ASCII terminal by providing the "look and feel" of a workstation, and the flexibility of networking.

An X station is a terminal that allows you to use the X Window System in a networked environment. You can network your X stations together with any number of host servers so that users can share all of the devices and computing power on the network. Users of X stations can access any application on any host that is compliant with the X Window System. The X station relies entirely on a UNIX system to provide it with the software resources and application programs. Why buy an X station instead of a second workstation? The appeal of X stations is their low cost relative to the cost of workstations, and the reduced cost-per-seat of installation and administration.

The X Window system is a portable, network-transparent, graphics-oriented window system developed at Massachusetts Institute of Technology (MIT) that runs under a wide variety of operating systems and hardware platforms.

© 2003 Hewlett-Packard Development Company, L.P.

The X Window system is a typical client/server application. Client and server processes can run on the same or on different network nodes because they use a network protocol stack for communication:

X server — The X server controls the interface between programs, input devices, and output devices. It coordinates input from devices such as the keyboard and the mouse or input from other applications. It is also responsible for updating the display (whether it be creating and manipulating windows or producing text and graphics).

X client — An X client is an application program that utilizes the services provided in the X environment. The X client communicates to the X server through the X protocol. The X protocol is a network-based protocol using an asynchronous byte-stream transfer. The interaction between the X server and an X client is characterized by the client's requests to the server, and the server's notification to the client of events such as a mouse button being pressed. A single X server can control many clients.

Nonclients — Nonclients must run within a terminal emulator because they cannot interact directly with the X server. They do not understand the X environment because they do not know the X protocol.

NOTE: On an X station with remote X clients, the X server is running locally and the X clients are running on the X session server or other remote nodes.

Hewlett-Packard supplies RISC-based X stations (with an Intel RISC processor) for both technical and commercial markets:

ENVIZEX — These stations are most often used with applications requiring higher display performance or expandability. ENVIZEX provides

- a floppy drive for easy transfer of files between UNIX and DOS
- a SCSI interface for connecting an HP ScanJet, thereby allowing for scanning documents
- an audio support capability with CD quality output
- a parallel and a serial (RS232) interface for printers

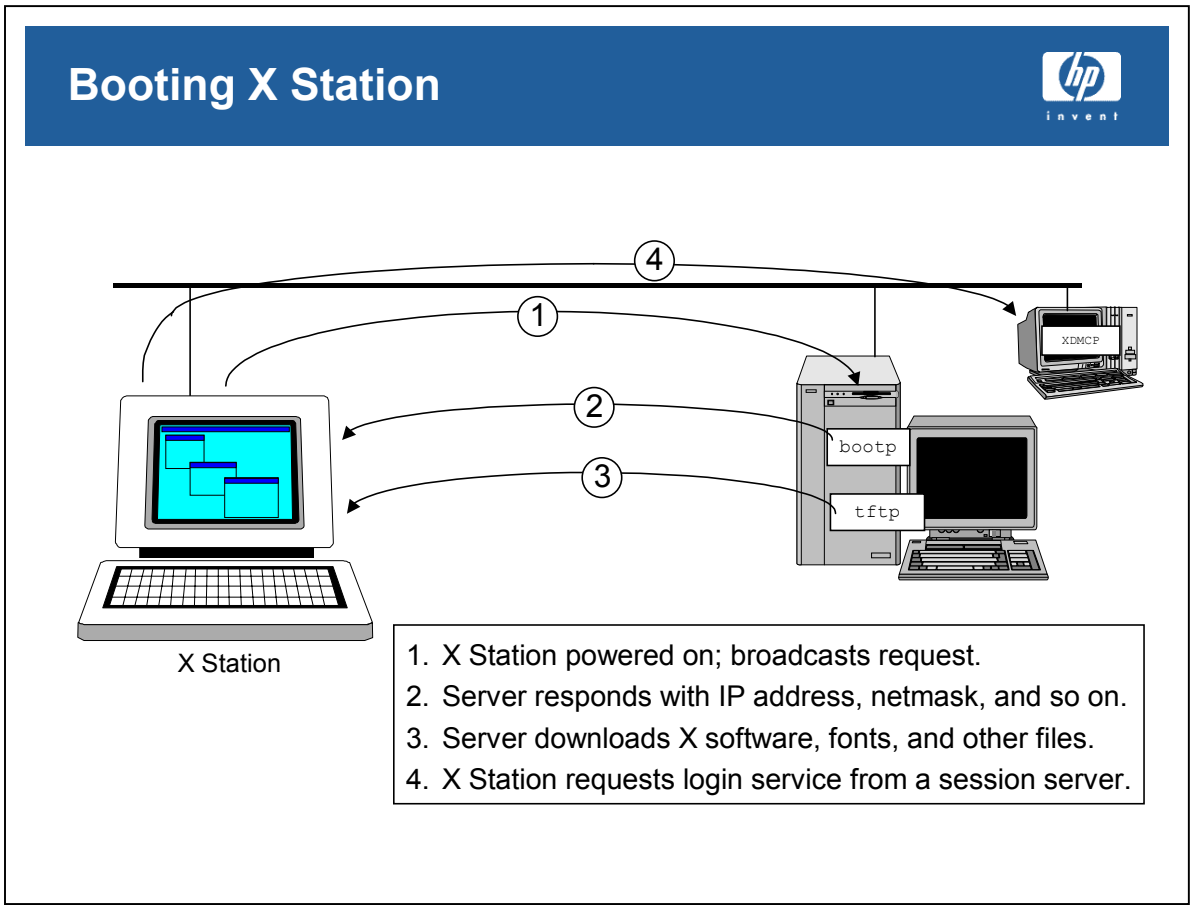
ENTRIA — These X stations are the lowest cost and simplest to manage graphical desktop. They are specifically designed for the commercial customer who currently uses ASCII terminals. ENTRIA is the ultimate terminal upgrade. It provides

- a parallel and a serial (RS232) interface for local print device support
- an optional audio card with CD quality output

ENWARE — This X station software provides a common software platform for ENVIZEX and ENTRIA as well as older 700/RX stations.

© 2003 Hewlett-Packard Development Company, L.P.

14–7. SLIDE: Booting X Station



Student Notes

After the X station is assembled, connected to the LAN, and powered on (and after the boot and file server has been configured to support the X station), the X station should be able to boot using **`bootp`** and **`tftp`**. Below is a list of what occurs at bootup.

1. The X station will send out a boot request on the LAN. The X station uses the boot request to ask any computer on the network to send the X station's network configuration parameters.

2. A host that has the **`bootpd`** daemon and an entry for the X station in its **`/etc/bootptab`** file, will respond to the X station's broadcast.

3. The server then uses **`tftp`** to send the X station the network information and its remote configuration file.

   Once the X station has its network parameters, it makes a connection with the file server (either using **`tftp`** or **`nfs`**) and downloads the X server code and fonts. Usually the boot and file server are the same computer.

© 2003 Hewlett-Packard Development Company, L.P.

4. The X station then directs a request using the XDMCP protocol by default to the file server asking for an X session manager login screen.

   After about 30 seconds, a login window appears.

If you have an account on that machine, you can now log in.

NOTE:	A ROM-based X station loads the X server and the fonts from the locally installed flash ROM card. A boot server is not needed if the network configuration parameters are configured locally in the X station's EEPROM.

Selecting an X Session Server with XDCMP

To access the X station's configuration screens, power on the X station, press the Space bar to abort the boot sequence, and press `To Configuration`.

The XDMCP request to find and select an X session server can be configured in the Server configuration screen. Press `Server`.

You have several choices to select an XDMCP request type:

Broadcast	The X station broadcasts its XDMCP request on the LAN interface (and the serial line, if configured for SLIP). All systems that will respond are displayed on a list. You can select the X session server you want.
Direct	The XDMCP request is sent to a specific system. The IP address of this host is specified in the Host field. If no IP address is specified, the file server's IP address is used.
Disabled	You need to start your X session clients manually, typically by using `Telnet` in the Terminal screen.
Indirect Request	The Host field must contain the IP address of the primary X session server.

With the `Server` screen, you can set up some more configurations:

Access Control	When enabled, this limits the systems that can access your display to those on the access control list in the X station's RAM. The default access control list contains the X session server (the Telnet host, if used) and the hosts listed in **/opt/enware/xthome/etc/ *X_stations_ip_address*.hosts**. The X station reads this file only on power up. The **xhost** command is a utility that changes the access control list stored in the X station's RAM for the current session. Changes with **xhost** disappear when you switch off the X station or when you log out.
X Server from	When you have an ROM-based X station, you can select `ROM` to load the X server from the flash ROM card.

© 2003 Hewlett-Packard Development Company, L.P.

RGB file from	When you have a ROM-based X station, you can select **ROM** to load the color database (**`rgb.txt`**) from the flash ROM card.
ROM Fonts	When you have a ROM-based X station, you can select **ROM** to load the font files from the flash ROM card.

NOTE: To get help, place the cursor on a configuration field and click the right mouse button. If this does not work, you can enable this feature in the subscreen **Preferences o** with the button **Click for Help**.

NOTE: When the X station is booted and you are logged in, you can get the configuration screens by pressing and holding **F12** for two seconds.

The X station's Network Configuration Screen

To access the X station's configuration screens, press **F12** and hold it for two seconds while you are logged in or abort the reboot of the X station by pressing the **Space** bar, then **To Configuration** .

The network configuration screen allows you to configure the required network parameters manually. In this case, you are independent from a boot server.

You can specify the following information:

Network Parameters	If you want to use a boot server you must choose **Download From Host (BOOTP)**; otherwise, choose **From Fields Below** so that your manual configuration is used.
IP Address	Enter the IP address of the X station in this field.
Subnet Mask	This is only necessary if subnetting is used in your network environment.
Gateways...	Specify the gateways (routers) if you want to access file servers and X session servers that do not belong to your network or subnet. Only one default gateway is allowed. Designate the default gateway with a **`Route to`** of **`0.0.0.0`**.
Setup SLIP..	This allows you to configure your X station for a Serial Line Internet Protocol (SLIP) session.
File Server	Specify the IP address of your file server for downloading the X server code, the font files, and the color database. You must also select the transfer method, TFTP or NFS. If the selected transfer method fails, the X station attempts to use the other access method.
Alt. File Server	This is used to specify an alternate file server if the primary file server is not available. You must also select the transfer method **TFTP** or **NFS**.

© 2003 Hewlett-Packard Development Company, L.P.

File Timeout	Specifies the length of time the X station attempts to access the primary file server before it tries to access the alternate file server. A large **`/etc/hosts`** table requires a longer time out. Allow 5 seconds for every 200 entries. The recommended solution is to list the X station near the top of the hosts table.
Name Server	If you use DNS (BIND), use this field to specify the IP address of your name server.
Alt. Name Server	Specify an alternate name server that will be used if the first one is not available.
Domain Name	If you use a name server, enter the X station's domain name in this field.
Terminal Name	This field contains the network node name of your X station.

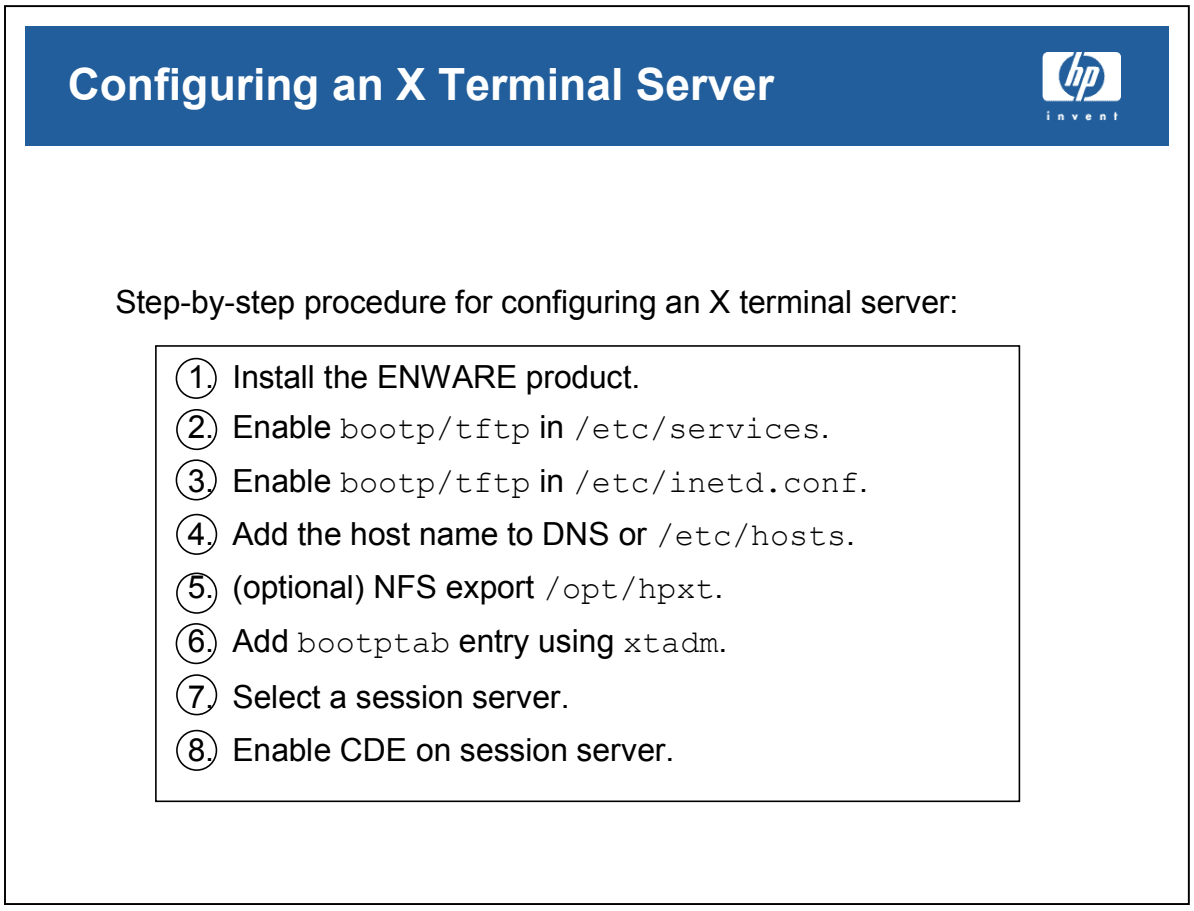
NOTE:	Don't forget to save your configuration.

NOTE:	To get help, place the cursor on a configuration field and click the right mouse button. If this does not work, you can enable this feature in the subscreen, Preferences o, with the **Click for Help** button.

NOTE:	You can use a password to restrict the access to the Network o screen and other screens. Configure the password in the Terminal o screen.

© 2003 Hewlett-Packard Development Company, L.P.

14–8. SLIDE: Configuring an X Terminal Server



Student Notes

Before you begin, you will need to know the following:

- IP address of the X station
- link level address of the X station
- X station's network node name
- subnet mask, if subnetting is used
- gateway IP address, if the there is a gateway between the X station and its X session server (or file server)
- name server IP address, if BIND is used

Most configuration steps can be performed with the utility **`/opt/hpxt/enware/bin/xtadm`** and SAM.

To manually configure a boot and file server for an X station, perform the following steps:

1. Install the ENWARE X station software with **`swinstall`**.

   With HP-UX 10.0, ENWARE 5.3 is needed. The software is installed to the **`/opt/enware`** directory and a symbolic link is created from **`/usr/lib/X11/700X`** to the

© 2003 Hewlett-Packard Development Company, L.P.

/opt/enware/xthome directory. In prior releases, **/usr/lib/X11/700X** was the location for all files.

2. Check **/etc/services** for bootp and tftp services.

   Both services must be registered in the **/etc/services** file. Add entries, if necessary.

```
/etc/services
+---------------------------------------------
|bootps  67/udp  # Bootstrap Protocol Server
|bootpc  68/udp  # Bootstrap Protocol Client
|tftp    69/udp  # Trivial File Transfer Protocol
```

3. Enable **bootpd** in **/etc/inetd.conf**.

```
# vi /etc/inetd.conf
+-----------------------------------------------------
|bootps dgram udp wait root /usr/lbin/bootpd bootpd

# inetd -c
# netstat -a | grep bootps
```

4. Configure and enable **tftpd** or set up the system as an NFS server for downloading files (X server code, fonts, and so on) to the X stations.

   — Configure **/usr/lbin/tftpd**.

```
# vi /etc/inetd.conf
+-------------------------------------------------------
|tftp dgram udp wait root /usr/lbin/tftpd tftpd \
| /opt/enware /usr/lib/X11/700X

# inetd -c
# netstat -a | grep tftp
```

   — Configure NFS server functionality and export the **/opt/enware** directory to the X—stations.

   You can use SAM to perform this task:

```
# sam
   |
   V
Networking/Communications ->
   |
   +--> Network Services --> NFS Server Enabled
   |
   +--> Networked File Systems ->
              |
              +--> Exported Local File Systems
```


© 2003 Hewlett-Packard Development Company, L.P.

The manual steps are as follows:

```
# vi /etc/rc.config.d/nfsconf
+-------------------------
| NFS_SERVER=1
| START_MOUNTD=1

# /sbin/init.d/nfs.server start

# vi /etc/exports
+-----------------------------------
|/opt/enware -access=xterm1

# exportfs -a
# exportfs
```

5. Add the X station's IP address and network node name to your name services' database (**/etc/hosts**, NIS, or DNS (BIND)).

6. Configure **/etc/bootptab** and create a configuration file for the X- station.

```
# vi /etc/bootptab
+-----------------------------------
|xterm1:\
|        ba:\
|        hn:\
|        ht=ether:\
|        ha=0800099856da:\
|        ip=192.6.1.108:\
|        bf=/opt/enware/xthome:\
|        gw=192.6.1.58:\
|        ds=192.6.1.11:\
|        sm=255.255.255.0:\
|        T144="xterm1.cfg":\
|        vm=rfc1048

# cd /opt/enware/xthome/config
# ll | more
# cp x_station_id.cfg xterm1.cfg
```

7. Test your installation with the **xtadm** utility.

   With **xtadm** you can test and check

   - the software installation
   - the operation of **bootpd**
   - the operation of **tftpd**
   - the operation of NFS

   See the **xtadm** screen listed below.

© 2003 Hewlett-Packard Development Company, L.P.

8. Use **/var/adm/inetd.sec** for security (optional).

   For network security, you can *allow* or *deny* the **bootps** and **tftp** services for specific nodes in the network. Typical entries are

```
# vi /var/adm/inetd.sec
+----------------------------
|bootps allow 192.6.1.*
|tftp allow 192.6.1.*
```

Configuration Screens of xtadm

The **/opt/hpxt/enware/bin/xtadm** utility is a user-friendly way for you to add or remove an X station, and to test the X station software installation (the operation of **bootp**, **nfs**, and **tftp**). **xtadm** can be invoked from the command line or through SAM's main menu (ENWARE X station Administration).

```
# /opt/hpxt/enware/bin/xtadm

     -------------------------------------
     HP X station ADMINISTRATIVE TASKS
     MAIN MENU



1) Add an X station

2) Remove an X station

3) Printers, plotters

4) Installation testing and version control

5) XDM Administration


    ?) Help     x) exit


Please enter selection (default=1): 4

     -------------------------------------
     HP X STATION ADMINISTRATIVE TASKS
     INSTALLATION TESTING MENU

1) Check software installation

2) Check operation of bootp

3) Check operation of tftp

4) Check operation of NFS
```


© 2003 Hewlett-Packard Development Company, L.P.

```
5) Manage software versions (list or change)

    ?) Help      p) previous menu

Please enter selection (default=1):
```

Configuring an X Session Server

There are two X session managers that are available on HP-UX systems, which can be used by X stations:

- CDE (Hewlett-Packard's Visual User Environment): **/usr/dt/bin/dtlogin**
- XDM (X Display Manager): **/opt/enware/xdm/xdm**

X session managers take care of the login process, check the password file, select a system shell, and set up the user's environment. They start X clients automatically and close the session when a user logs out. CDE and XDM differ in the user interface they provide, the session configuration files they use, and the amount of host resources each requires. Generally, CDE is more user-friendly, but requires significant host resources. XDM provides a Motif environment that may require the user to be more UNIX-literate, but it uses fewer host resources than CDE.

NOTE:	The X session manager subsystem is usually started at boot time. **xdm** and **dtlogin** should not run at the same time.

CDE is automatically started by the **/sbin/init.d/dtlogin.rc** script at run level 3 of the system startup procedure.

To provide the X station XDM, an **xdm** daemon has to run. The **xdm** daemon can be configured with the **xtadm** utility. XDM software is not bundled with the operating system, it is part of the ENWARE software and must be installed with **swinstall**. An **xdm** login uses **$HOME/.xsession** as the session startup file to start the window manager and clients. If **.xsession** is not found, /**opt/enware/xdm/sys.xsession** is used instead. Error messages from clients, if any, are put in the **$HOME/.xsessionlog** file.

The following is an example for an XDM entry in **/etc/inittab**:

```
/etc/inittab
+-------------------------------------------------------------------------
|init:3:initdefault:
|   :
|   :
|xd:234:respawn:/opt/enware/xdm/xdm -nodaemon < /dev/null > /dev/null 2>&1
```


© 2003 Hewlett-Packard Development Company, L.P.

14–9. LAB: Managing a bootp/tftp Server

Part 1: Basic bootp/tftp Configuration

1. The **bootp**/**tftp** services are bundled in the InternetSrvcs product. Ensure the InternetSrvcs product is installed on your machine.

```
# swlist -l product InternetSrvcs
```

2. Ensure that the **bootps** and **tftp** services are both enabled in **/etc/inetd.conf** and the **/etc/services** file.

```
# more /etc/inetd.conf  # Are tftp and bootps commented in?
# more /etc/services    # Are tftp and bootps commented in?
# inetd -c              # Reread inetd.conf if necessary
```

3. Files available for download via **tftp** are often stored in a **tftp** home directory. If you don't already have a **tftp** account in your **/etc/passwd** file, create it from the SAM Network services screen.

```
SAM --> Networking and communications --> Network services
        (select TFTP)
        Actions --> Enable
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring `bootp/tftp` Service for a Network Printer

1. Using **`hppi`**, create a **`bootptab`** entry for a network printer. Use the hardware address, **`IP address`**, host name, subnet mask, and default router address provided by your instructor. Use your classroom's room name or number as the **printer location**, and your own name as the printer **contact**.

   ```
   # /opt/hpnpl/bin/hppi
     -> (2) JetDirect Configuration
     -> (1) Create printer configuration in BOOTP/TFTP database
   ```

   (Answer the questions that follow according to the instructor's directions.)

NOTE:	The **`hppi`** utility may complain that a "network port is being used by **`rbootd`**." Despite this warning, **`hppi`** should successfully configure the printer. The warning can be ignored.

2. Check the **`/etc/bootptab`** file for changes made by **`hppi`**. Name three pieces of information defined in the printer's new entry in **`bootptab`**.

3. At this point your machine is ready to service **`bootp`** requests from the network printer you configured.

4. Now remove the new printer **`bootp`** configuration from your machine using **`hppi`**.

   ```
   # /opt/hpnpl/bin/hppi
     -> (2) JetDirect Configuration
     -> (2) Remove printer configuration from BOOTP/TFTP database
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring bootp/tftp Service for an X-Terminal

1. Hosts that manage X terminals must have the "Enware" product installed. Check to ensure that your machine has "Enware" installed.

```
# swlist -l product ENWARE
```

2. Some X terminals download the Xserver software via **tftp**, while others use **nfs**. Ensure that your host is configured as an NFS server, and that the **/opt/hpxt** directory is included in your **/etc/exports** file.

```
# ps -e | grep nfsd       # nfs server process should be running!
# exportfs                # /opt/hpxt should be exported
```

3. Now configure an X terminal entry in your **/etc/bootptab** file using **xtadm**. Use the MAC, IP, host name, subnet mask and default router address suggested by your instructor.

```
# /opt/hpxt/enware/bin/xtadm
       --> (1) Add an X station
 (answer q's that follow using addresses provided by instructor)
```

4. Look at the **/etc/bootptab** entry that was created for your X terminal. List three pieces of information defined for your X terminal in the new **bootptab** entry.

5. X terminals download a configuration file from the **bootp**/**tftp** server. These configuration files are normally stored in **/opt/hpxt/enware/xthome/config/**. Your new X terminal should have an entry in this directory. Search through the file for the

© 2003 Hewlett-Packard Development Company, L.P.

lines labeled "Startup Session" and "XDMCP Host." These two lines determine the session server that your new X terminal will connect to.

By default, the X terminal will connect to the **bootp/tftp** server as the session server as well. You can specify another session server by adding the following lines anywhere in the configuration file:

```
Startup Session = XDMCP Direct
XDMCP Host      = host name_of_session_server
```

Configure your new X terminal to connect to the session server named by your instructor.

6. Now your host is ready to provide **bootp/tftp** server to your newly defined X terminal. If the X terminal you defined is available in the classroom, reboot it to see if your configuration was successful.

7. Remove the **bootp** entry you added for your X terminal using **xtadm**.

```
/opt/hpxt/enware/bin/xtadm
          -->(2) Remove an X Station

(answer the questions that follow)
```


© 2003 Hewlett-Packard Development Company, L.P.

Module 15 — Configuring NTP

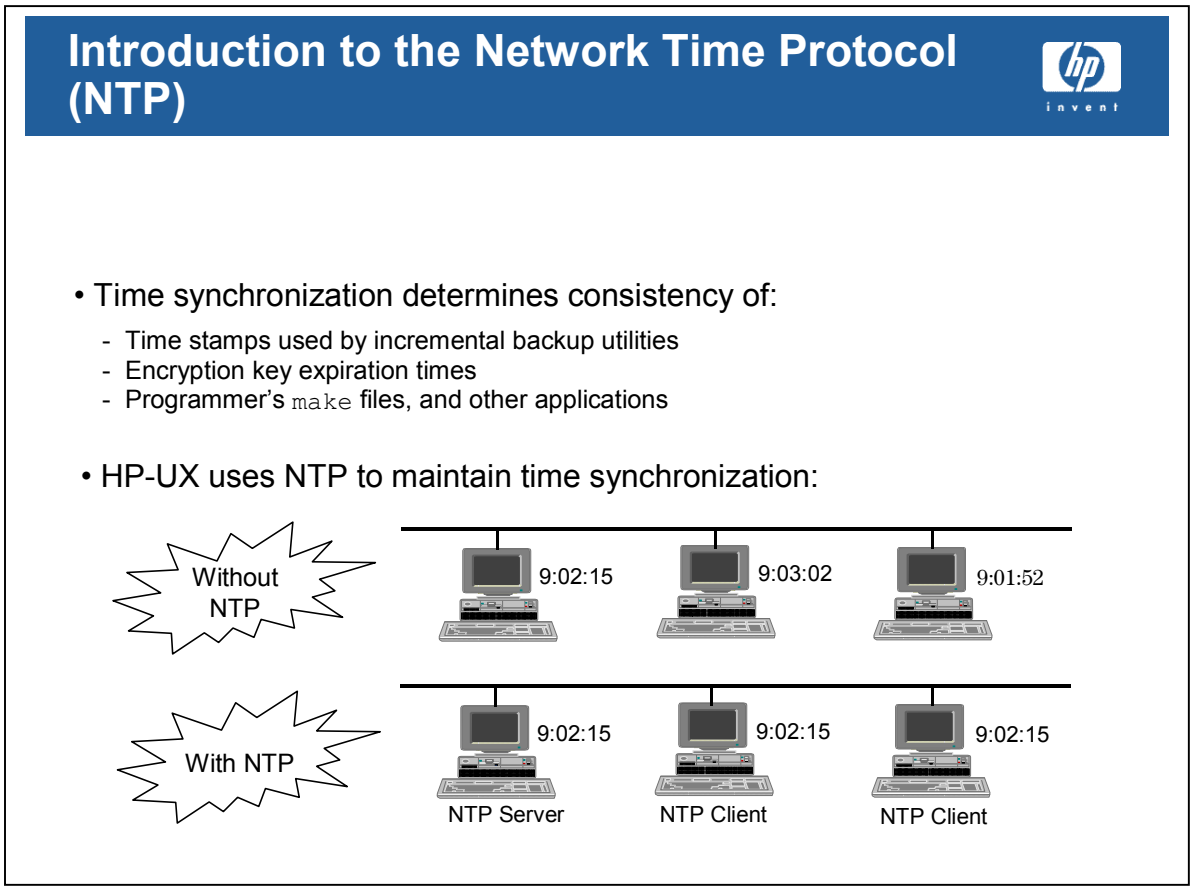
Objectives

Upon completion of this module, you will be able to do the following:

- List three reasons for implementing network time synchronization.
- Describe the NTP **stratum level** concept.
- Define the following terms:
  - NTP server
  - NTP peer
  - NTP broadcast client
  - NTP polling client
- Configure an NTP server.
- Configure an NTP broadcast client.
- Configure an NTP direct-poll client.
- Monitor NTP using the **`ntpq`** command.

© 2003 Hewlett-Packard Development Company, L.P.

15–1. SLIDE: Introduction to the Network Time Protocol (NTP)



Student Notes

Many computer applications rely on the system clock to accurately determine the current system time.

- System backup utilities use the system clock and file time stamps to determine which files should be included in incremental backups.
- More and more security sensitive organizations are using Kerberos or other authentication/encryption mechanisms to protect their data. These security tools often use authentication keys that expire after a period of time. In order for this mechanism to function properly, the system clock must be accurate!
- Programmers oftentimes use the **`make`** utility to compile and link programs. **`make`** depends on the system clock and file time stamps to determine when source code files have changed.

In large, networked environments where hosts share files and other resources, it is critical that hosts maintain accurate, or at least consistent, time to avoid causing problems for the time-sensitive applications listed above. Humans rarely notice a discrepancy of one or two seconds between hosts, but time-sensitive applications might!

© 2003 Hewlett-Packard Development Company, L.P.

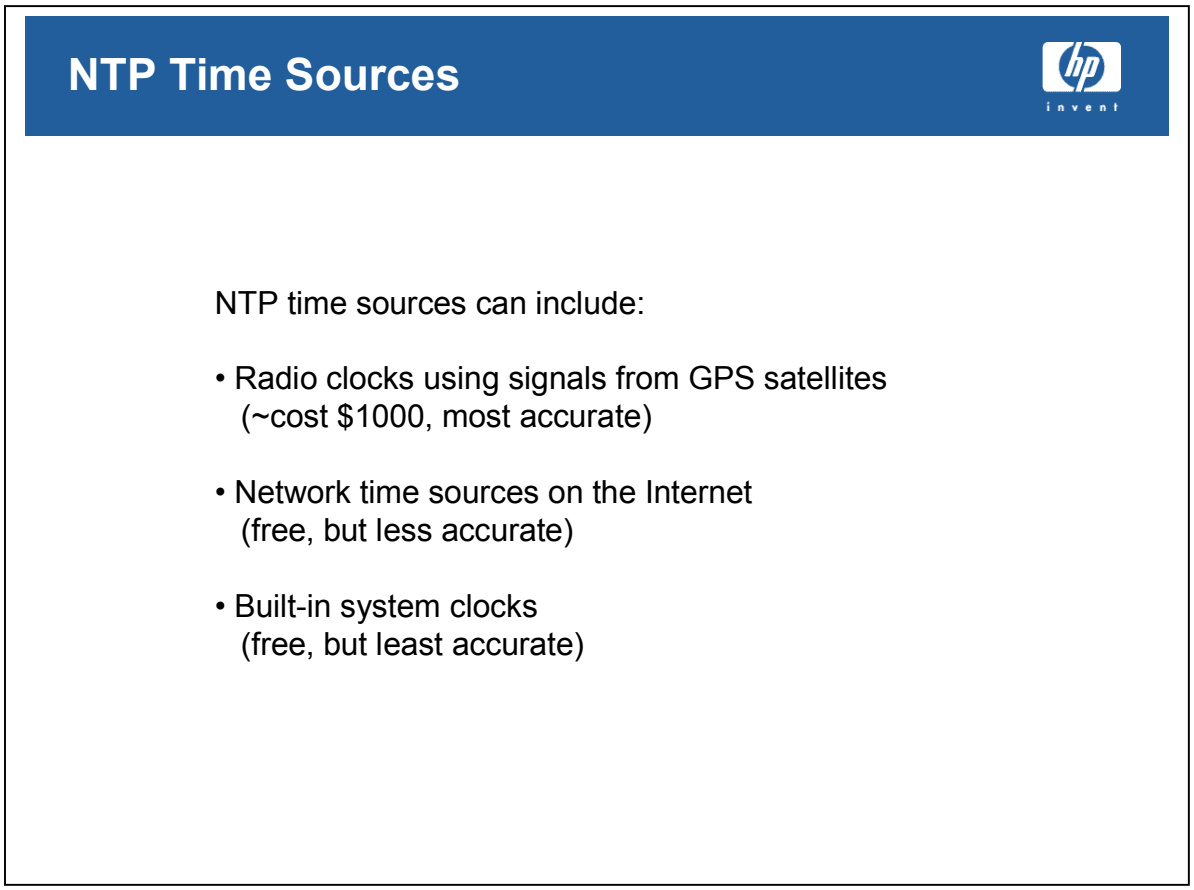
Unfortunately, the built-in clocks in today's computers are not perfect. Even the best system clocks may gain or lose a second or two per day. In order to ensure consistent time stamps across their LANs, many administrators choose to synchronize their hosts' system clocks using the Network Time Protocol, or NTP.

NTP was developed at the University of Delaware, and is bundled with HP-UX. The HP-UX **`xntpd`** daemon is used to implement the NTP service in HP-UX.

NTP is configurable through the command line or through SAM.

© 2003 Hewlett-Packard Development Company, L.P.

15–2. SLIDE: NTP Time Sources



Student Notes

NTP can be used to synchronize system clocks using a variety of time sources:

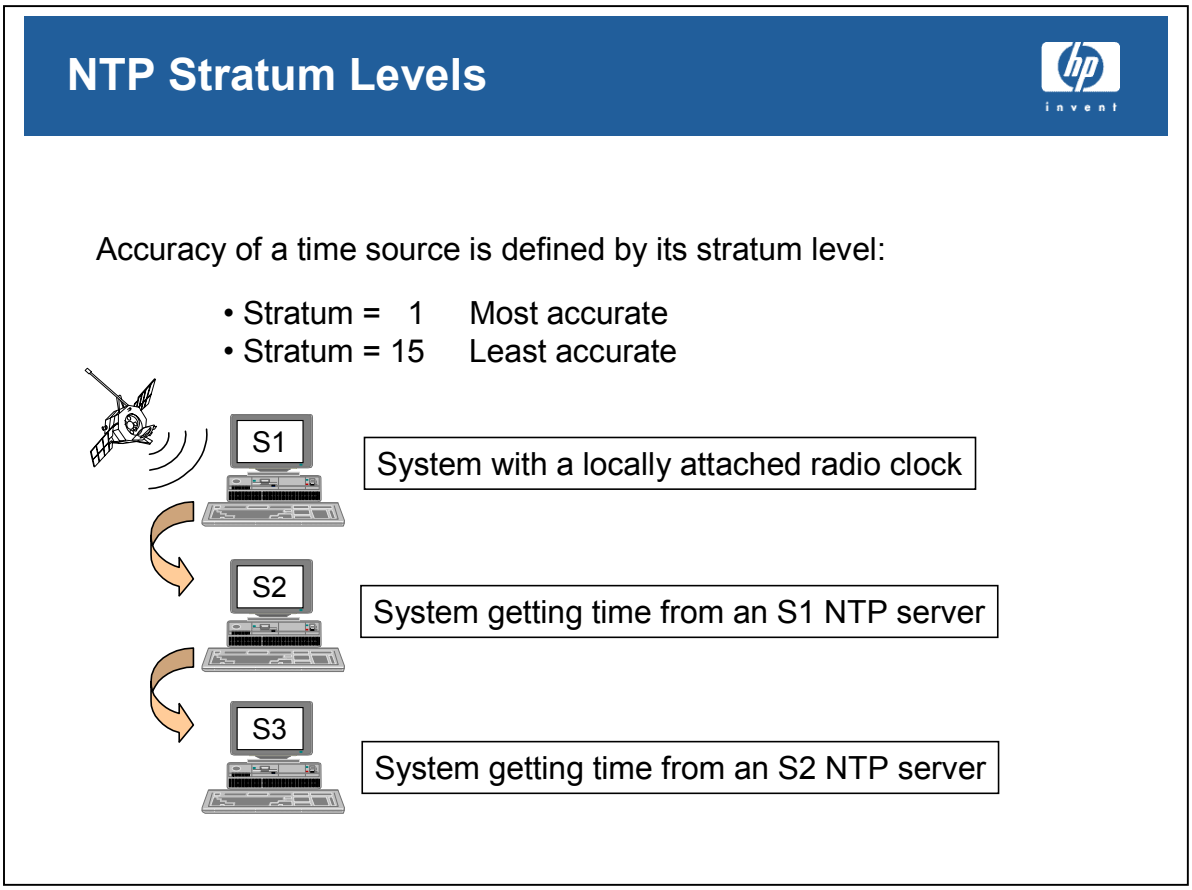
- A radio clock can be attached to the serial port of an HP-UX system. A radio clock determines the current time using signals from GPS (Global Positioning System) satellites or other radio time sources. Radio clocks are among the most accurate time sources, but cost several thousand dollars. A list of radio clock suppliers is available at **`http://www.ece.udel.edu/~ntp`**. Before purchasing a clock, verify that the model you choose is supported by HP.

- If you cannot afford a radio clock, a public NTP timeserver on the network can be used to synchronize a system's clock. A list of public NTP timeservers on the public internet is available from **`http://www.ece.udel.edu/~ntp`**.

- If you do not have a radio clock or an Internet connection, select one host on your local network as your "authoritative" time source. Other nodes on the LAN, then, can synchronize their system clocks to the selected "authoritative" source. This guarantees that hosts on your LAN agree on a common system time, but does not guarantee that your hosts are synchronized with other hosts outside your local network.

© 2003 Hewlett-Packard Development Company, L.P.

15–3. SLIDE: NTP Stratum Levels



Student Notes

NTP Stratum Levels

In a large network, several hierarchically organized timeservers can be used to synchronize the clocks of all systems on the network. Every network time source is assigned a "stratum level," which reflects the time source's accuracy.

Hosts with directly connected radio clocks are considered stratum 1 time sources. Timeservers that obtain the system time by polling a stratum 1 server across the Internet are typically considered stratum 2 servers. Servers that obtain the system time from stratum 2 servers are typically considered stratum 3 servers. Thus, servers with lower stratum levels are likely to be more accurate time sources.

NTP Network Delay

Note, however, that a server's stratum level is not the only parameter that affects the quality of a time source. Network delay is often a critical factor to consider when choosing a time source. Collisions, routers, and heavy network traffic can all dramatically affect the quality of time service available from an NTP server.

© 2003 Hewlett-Packard Development Company, L.P.

Choosing an NTP Time Source

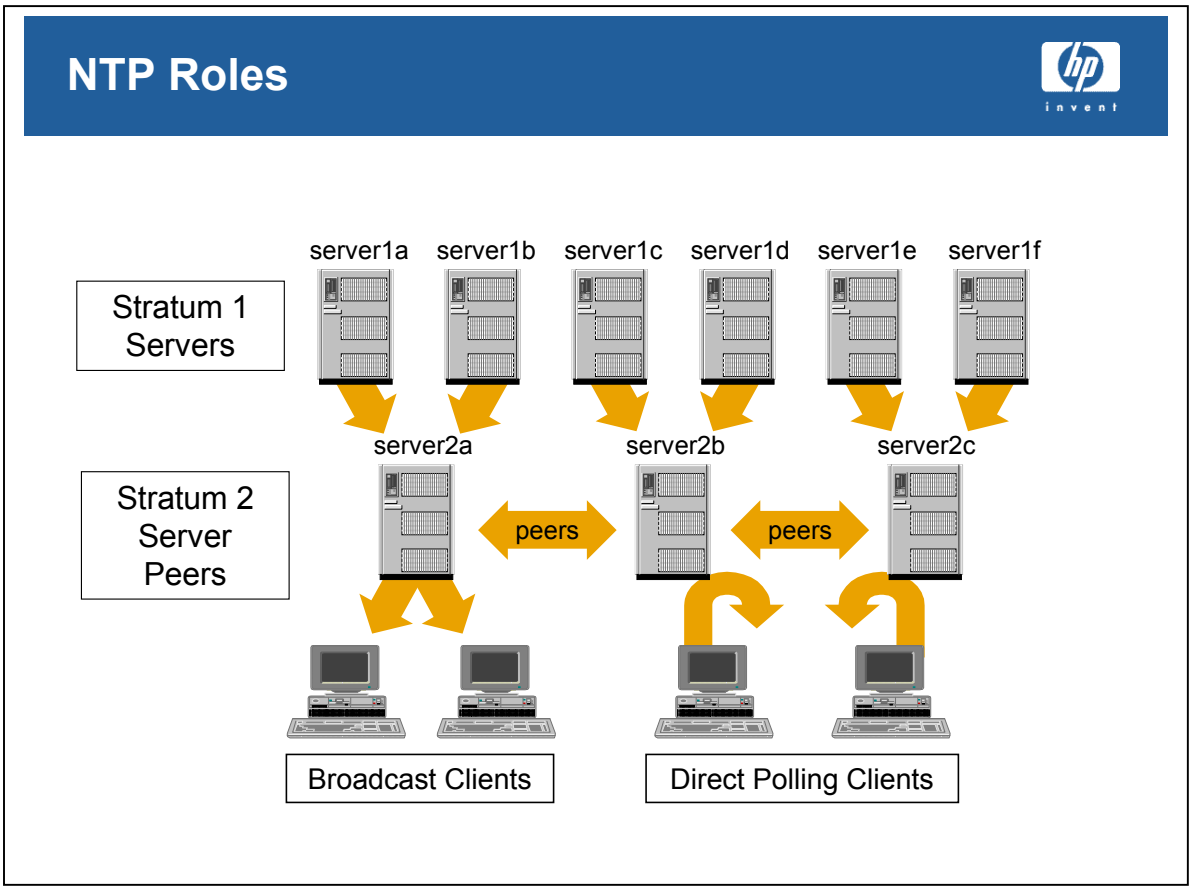
When choosing an NTP timeserver, start by consulting the University of Delaware web page NTP server list. Then **`ping`** a few servers, and choose the server with the best round trip **`ping`** travel time. Servers that yield **`ping`** values greater than 500ms should be avoided.

NTP Etiquette

Before you configure your **`xntpd`** daemon to access a public NTP timeserver, check the University of Delaware web page to see if the server administrator requires some sort of registration, or imposes any restrictions on NTP clients. Ideally, you should configure two or three NTP servers on your local network to poll a stratum 1 or 2 server on the Internet, then configure other hosts on your local network to poll these local NTP servers. This minimizes the load on the public timeservers.

© 2003 Hewlett-Packard Development Company, L.P.

15–4. SLIDE: NTP Roles



Student Notes

When implementing NTP on a network, systems can play four possible roles:

NTP Servers	An NTP server provides time service to other systems.
NTP Peers	Many NTP servers form peer relationships with other same-stratum servers. If a stratum 2 server loses connectivity to its stratum 1 time source, it may temporarily use the time service provided by a stratum 2 peer.
NTP Direct Polling Clients	A direct polling client regularly polls one or more NTP servers, compares the servers' responses, and synchronizes the system clock to the most accurate time source.
NTP Broadcast Clients	An NTP broadcast client passively listens for NTP broadcasts from NTP servers on the local network. Broadcast clients generate less network traffic than direct polling clients, but provide less accuracy.

The example on the slide shows a typical NTP configuration. The servers at the top of the slide are stratum 1 servers on the Internet with locally attached radio clocks.

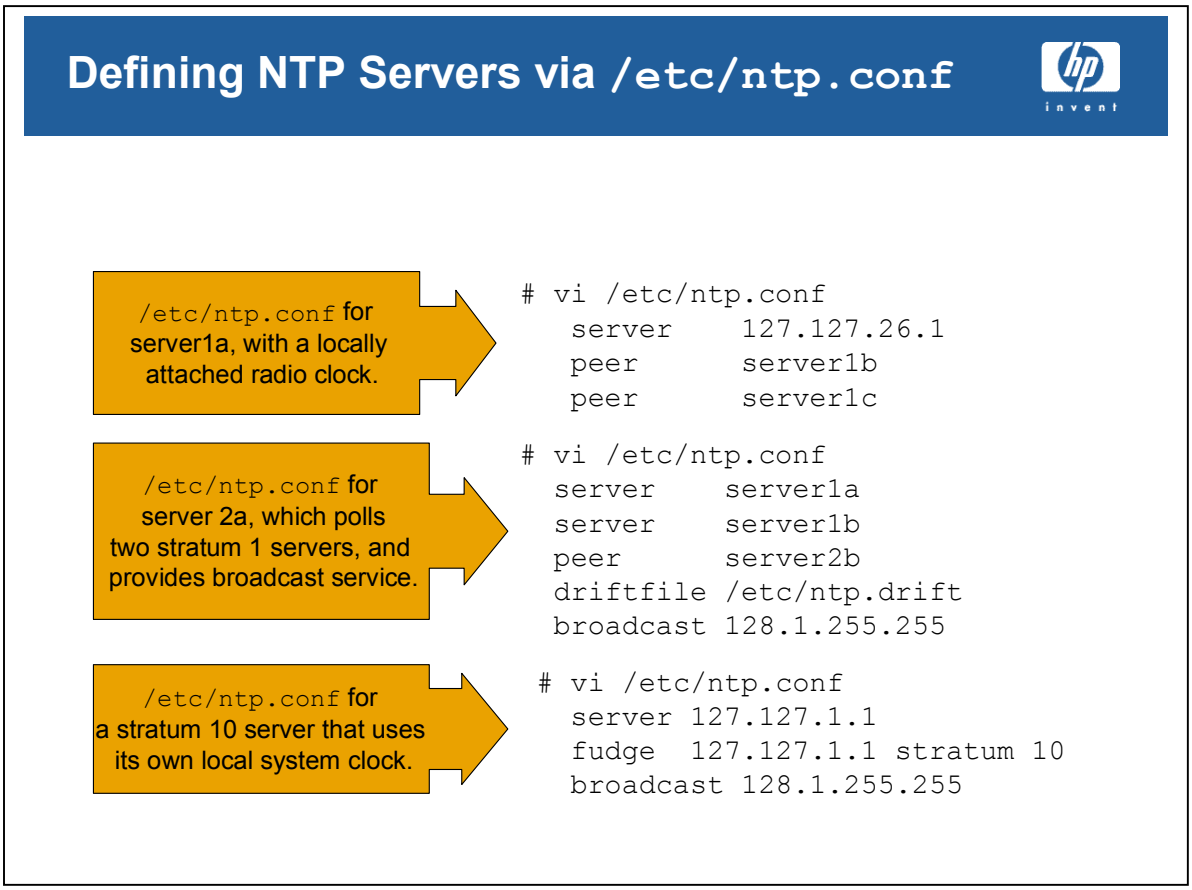
© 2003 Hewlett-Packard Development Company, L.P.

The second tier servers on the slide are stratum 2 servers that poll stratum 1 servers to obtain the current system time. It is recommended that each stratum 2 NTP server consult three or more stratum 1 servers to ensure reliability. The **`xntpd`** daemon will automatically poll both stratum 1 servers and synchronize to the source that it deems most accurate. To further improve reliability, each stratum 2 server should form a peer relationship with one or more other stratum 2 servers.

Finally, the slide shows two broadcast clients that passively listen for NTP broadcasts, and two direct polling clients that regularly poll their respective servers to obtain NTP service. If you have several NTP servers on your local network, you may choose to have your clients poll all of these servers to ensure reliability.

© 2003 Hewlett-Packard Development Company, L.P.

15–5. SLIDE: Defining NTP Servers via `/etc/ntp.conf`



Student Notes

The **`/etc/ntp.conf`** file is used to define a system's NTP relationships with other systems on the network. The file is read by the **`xntpd`** daemon during the system startup process.

Configuration for a Stratum–1 Server with a Radio Clock

To configure a stratum 1 server, add the following lines to the **`/etc/ntp.conf`** file (this sample file might be used by server1a in the example on the previous slide):

```
# vi /etc/ntp.conf
  server    127.127.26.1
  peer      server1b
  peer      server1c
```

Notes regarding the above entry:

- The 127.127.26.1 IP address is a pseudo IP address that **`xntpd`** uses to determine what type of radio clock is attached to your system. This particular address indicates that the system has an HP58503A GPS clock attached. Refer to the comment lines in **`/etc/ntp.conf`** for the pseudo IP addresses used by other clocks.

© 2003 Hewlett-Packard Development Company, L.P.

- Each radio clock server should peer with several other stratum-1 servers in case the local radio clock becomes unavailable. This sample file defines peer relationships with server1b and server1c.

Configuration for a Stratum–2 Server

Below is an example of NTP configured for stratum 2-server server2a from the previous diagram.

```
# vi /etc/ntp.conf
  server    server1a
  server    server1b
  peer      server2b
  driftfile /etc/ntp.drift
  broadcast 128.1.255.255
```

Notes regarding the above entry:

- The **`server`** entries determine which stratum 1 servers this server should poll to obtain time service.
- The **`peer`** entry defines a peer relationship with another stratum 2 server, server2b.
- The **`driftfile`** entry specifies the name of a file to use to track long-term drift of the local clock.
- The **`broadcast`** entry causes **`xntpd`** to regularly broadcast the official NTP time to broadcast clients on the 128.1.0.0/16 network.

Configuration for a Local NTP Server Using its Internal Clock

To configure an NTP server to use its own system clock as an authoritative time source, add the following lines to the server's **`/etc/ntp.conf`** file:

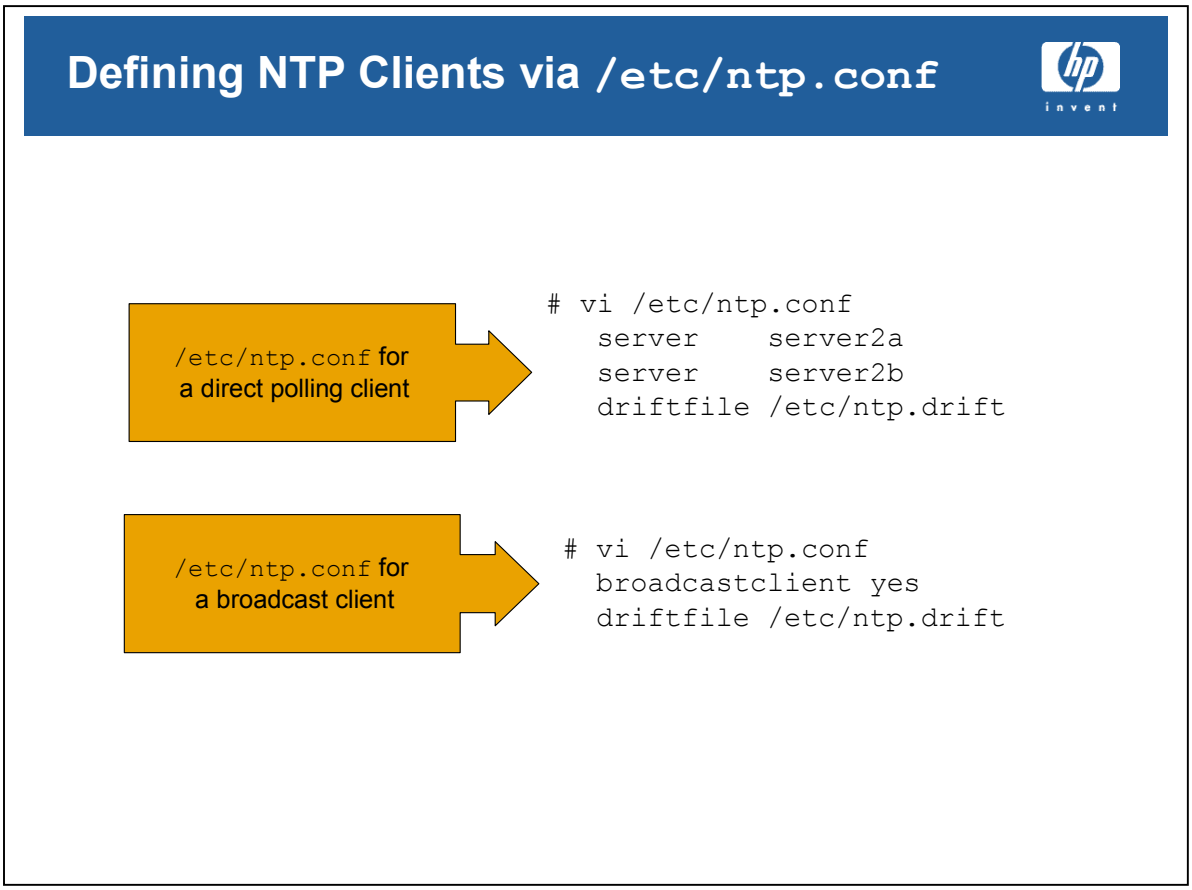
```
server 127.127.1.1 prefer
fudge 127.127.1.1 stratum 10
broadcast 128.1.255.255
```

Notes regarding the above entry:

- The IP address is a psuedo IP address that identifies the local system as a time source.
- The **`fudge`** entry defines a stratum level to be assigned to this clock. It is a good idea to treat the internal system clock as a stratum 10 time source so clients that have access to real NTP servers will synchronize to those servers.
- The **`broadcast`** entry causes the server to broadcast NTP information to broadcast clients on the 128.1.255.255 network.
- This method of time synchronization should only be used on networks with no access to an external time source.

© 2003 Hewlett-Packard Development Company, L.P.

15–6. SLIDE: Defining NTP Clients via `/etc/ntp.conf`



Student Notes

Each NTP client should have an **`/etc/ntp.conf`** configuration file, too.

Configuration for a Client using Direct Server Polling

To configure a client to poll a specific NTP server, add the following line to the client's **`/etc/ntp.conf`** file:

```
# vi /etc/ntp.conf
   server    server2a
   server    server2b
   driftfile /etc/ntp.drift
```

Notes regarding the above entry:

- The client will periodically poll server2a and server2b. The default polling interval starts at 64 seconds, but may increase over time. Each client should poll multiple NTP servers to ensure reliability.
- The **`driftfile`** is used to track differences between the client's time and the server's time. As the driftfile stabilizes, the server will be polled less frequently.

© 2003 Hewlett-Packard Development Company, L.P.

Configuration for a Client using Broadcast Polling

To configure a client to listen for time broadcasts, add the following line to the client's **`/etc/ntp.conf`** file:

```
# vi /etc/ntp.conf
  broadcastclient yes
  driftfile /etc/ntp.drift
```

Notes regarding the above entry:

- The client will passively listen for NTP broadcasts and adjust his clock appropriately.
- This method is recommended over direct server polling for large networks since it significantly reduces NTP network traffic.
- Clients must be on the same subnet as the NTP broadcast server.

© 2003 Hewlett-Packard Development Company, L.P.

15–7. SLIDE: How NTP Adjusts the System Clock

How NTP Adjusts the System Clock

`/usr/sbin/ntpdate -b server server server`

- Utility called once at system boot
- Polls one or more NTP servers
- "Steps" local clock immediately to match the most accurate server

`/usr/sbin/xntpd`

- Daemon started at system boot
- Polls one or more NTP servers at regular intervals
- "Slews" local clock gradually to match the most accurate server

`/etc/ntp.drift`

- File maintained and used by **xntpd**
- Tracks the local clock's accuracy over time

Student Notes

NTP provides three different mechanisms for synchronizing your system clock with other nodes on the network.

The `ntpdate` Command

The **`ntpdate`** command, when executed with the **`-b`** option, polls one or more NTP servers, then immediately "steps" the system clock to synchronize with the most accurate NTP server. This is the quickest way to get a client's clock in sync with the NTP server's time. However, stepping the system clock forward (or backward!) can wreak havoc on running applications. For this reason, most systems only execute **`ntpdate`** during system startup, before applications are launched.

The `xntpd` Daemon

After **`ntpdate`** initially synchronizes the system clock at boot time, the **`xntpd`** daemon runs continuously in the background, periodically polling the NTP servers defined in **`/etc/ntp.conf`**, and "slewing" the system clock as necessary to maintain synchronization. These small, gradual adjustments over time should be transparent to your applications. If the local clock ever diverges from the NTP time sources by more than 1000 seconds, the **`xntpd`** daemon assumes that the server has been corrupted, and dies.

© 2003 Hewlett-Packard Development Company, L.P.

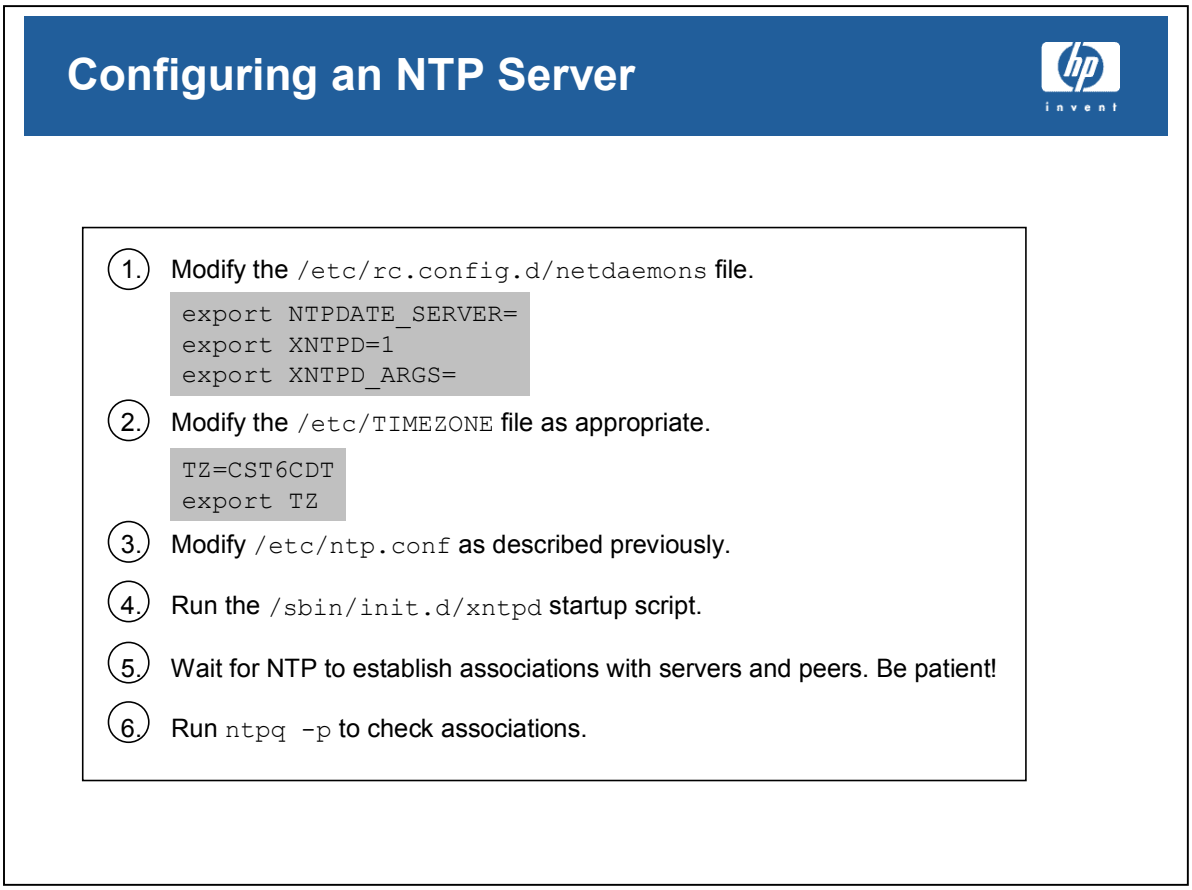
The `/etc/ntp.drift` File

A system's internal system clock will tend to be consistently fast, or slow, relative to the NTP timeservers. Over time, the **`xntpd`** daemon computes the internal system clock's average "drift," compensates accordingly, and polls the NTP servers less frequently. This minimizes NTP network traffic.

Configuring a **`driftfile`** entry in **`/etc/ntp.conf`** causes **`xntpd`** to record the internal system clock's average drift in the **`/etc/ntp.drift`** file. The **`driftfile`** allows **`xntpd`** to reestablish more quickly the system clock drift value after reboots.

© 2003 Hewlett-Packard Development Company, L.P.

15–8. SLIDE: Configuring an NTP Server



Student Notes

Several steps are required to configure an NTP server:

1. Edit the **/etc/rc.config.d/netdaemons** file to configure the **xntpd** daemon to startup every time the system boots. Set the **XNTPD** variable to equal 1.

   ```
   # vi /etc/rc.config.d/netdaemons
     export NTPDATE_SERVER=
     export XNTPD=1
     export XNTPD_ARGS=
   ```

   If the server uses a radio clock, or the internal system clock, leave the **NTPDATE_SERVER** variable null. If the server obtains its system time from other network timeservers, the **NTPDATE_SERVER** variable should be set equal to a space-separated list of timeservers.

2. Edit the **/etc/TIMEZONE** file and specify the correct time zone for the system. Set the TZ variable to equal the time zone for the system. See the **/usr/lib/tztab** file for a list of all the available time zones.

   ```
   # vi /etc/TIMEZONE
     TZ=CST6CDT
     export TZ
   ```


© 2003 Hewlett-Packard Development Company, L.P.

3. Edit the **/etc/ntp.conf** file and define the NTP server as described earlier in this module.

4. Start the **xntpd** daemon manually by executing the following command:

   ```
   # sbin/init.d/xntpd start
   ```

5. Wait. It could take up to 6 minutes for the **xntpd** daemon to stabilize.

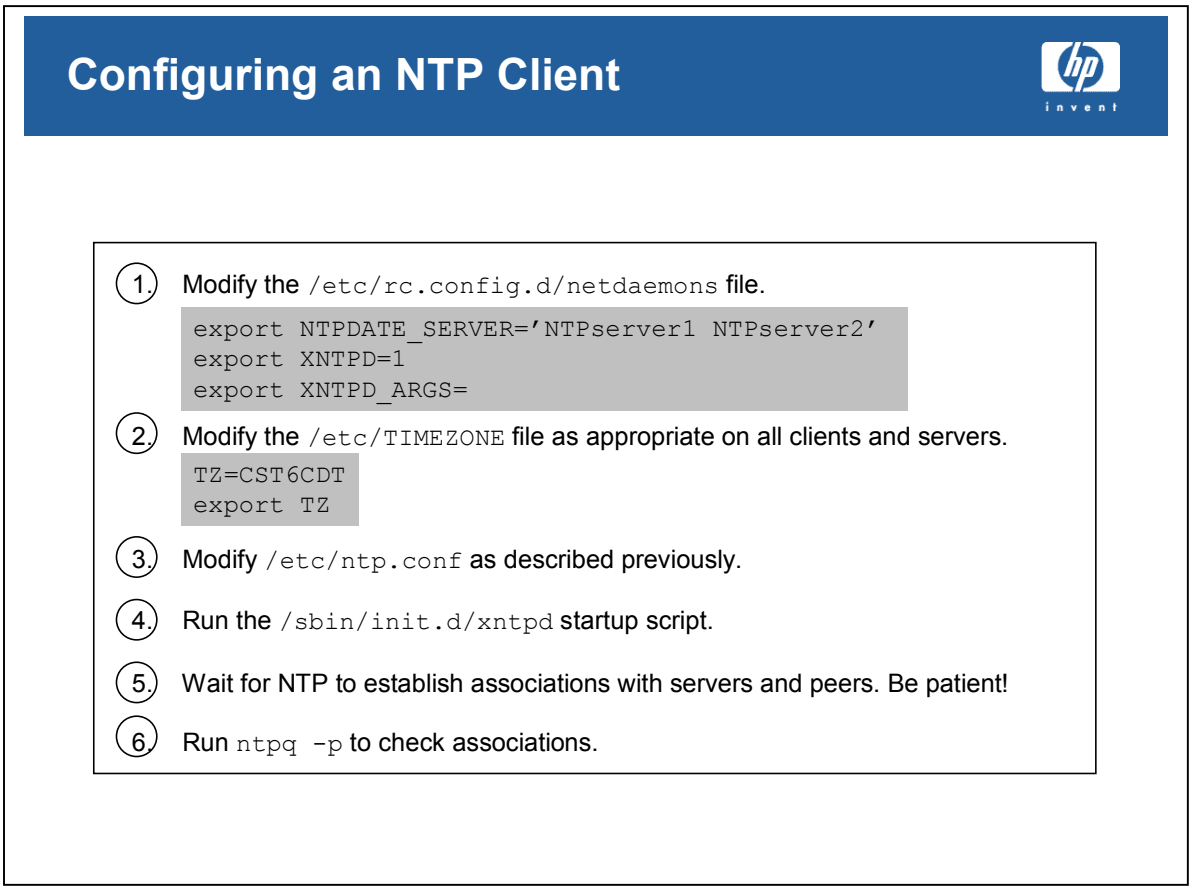
6. Verify the NTP server configuration (and its association with peer NTP servers) by executing the following command:

   ```
   # ntpq -p
   ```

   More information on the **ntpq** command is contained in the upcoming slides.

© 2003 Hewlett-Packard Development Company, L.P.

15–9. SLIDE: Configuring an NTP Client



Student Notes

The procedure for configuring an NTP client is virtually identical to that of configuring an NTP server — only the contents of the configuration files change.

The complete, step-by-step procedure for configuring an NTP client is:

1. Edit the **/etc/rc.config.d/netdaemons** file to configure the **xntpd** daemon to startup every time the system boots. Set the XNTPD variable to 1, and specify which NTP servers to query when the **ntpdate** command is used:

```
# vi /etc/rc.config.d/netdaemons
  export NTPDATE_SERVER='NTPserver1 NTPserver2'
  export XNTPD=1
  export XNTPD_ARGS=
```

2. Edit the **/etc/TIMEZONE** file and specify the correct time zone for the client system. See the **/usr/lib/tztab** file for a list of all the available time zones.

```
# vi /etc/TIMEZONE
  TZ=CST6CDT
  export TZ
```


© 2003 Hewlett-Packard Development Company, L.P.

3. Edit the **/etc/ntp.conf** file and define the NTP client as described earlier in this module.

4. Start the **xntpd** daemon manually by executing the following command:

   ```
   # /sbin/init.d/xntpd start
   ```

5. Wait for the **xntpd** daemon to start. It could take up to 6 minutes for the daemon to establish an association with its NTP servers and peers.

6. Verify association with NTP server(s) and peers were correctly established. Execute the command:

   ```
   # ntpq -p
   ```


© 2003 Hewlett-Packard Development Company, L.P.

15–10. SLIDE: Verifying NTP Functionality

Verifying NTP Functionality

- View NTP activity and errors over time:

```
# more /var/adm/syslog/syslog.log
```

- Verify that the `xntpd` daemon is running:

```
# ps -e | grep xntpd
```

- Check associations with other nodes:

```
# ntpq -p
  remote    refid    st t when  poll  reach  delay  offset   disp
  ------------------------------------------------------------------
  *server2a server1a  3 u  64    64    377   0.87   10.56   16.11
  +server2b server1b  3 u 100   264    376   9.89    5.94   16.40
   server2c 0.0.0.0  16 -   -    64      0   0.00    0.00 1600.00
```

Student Notes

Several tools are available to verify that NTP is functioning properly.

- Check the **`syslog.log`** log file:

```
# tail /var/adm/syslog/syslog.log
```

  When the **`xntpd`** daemon starts up, it logs a number of entries to the **`/var/adm/syslog/syslog.log`** log file, including:

  - Timestamps of when the **`xntpd`** was started and stopped.
  - Associations formed with other nodes running NTP.
  - Errors found in the **`/etc/ntp.conf`** file.

- Verify that the **`xntpd`** daemon is running by executing the **`ps`** command:

```
# ps -e | grep xntpd
```


© 2003 Hewlett-Packard Development Company, L.P.

- View the relationships established by your **xntpd** daemon by executing the **ntpq -p** command.

```
# ntpq -p
  remote    refid    st t when  poll  reach  delay  offset   disp
  ------------------------------------------------------------------
  *server2a server1a  3 u  64    64    377   0.87   10.56   16.11
  +server2b server1b  3 u 100   264    376   9.89    5.94   16.40
   server2c 0.0.0.0  16 -   -    64      0   0.00    0.00 1600.00
```

ntpq displays several fields of information for each of the defined NTP relationships. The fields are described below:

remote:	Identifies the NTP source's host name.
refid:	Where the NTP source obtained its time (0.0.0.0 indicates a downed server).
st:	Stratum level of the source (low is best!).
t:	Source type. l=local GPS, radio, or system clock; u=unicast; b=broadcast.
when:	How long has it been since the server responded to a poll?
poll:	How frequently is NTP polling the server?
reach:	A value of 0 means the server is unreachable; 377 means that all recent probes have been successful.
delay:	Milliseconds required for the server to reply to a query (low is best!).
offset:	Milliseconds difference between the this host and the server (low is best!).
disp:	How much does the network delay vary from poll to poll? (low is best!)

The NTP source that you are currently synchronized to is indicated by a "*". Other strong contenders are indicated by a "+". "-" indicates a discarded source.

© 2003 Hewlett-Packard Development Company, L.P.

15–11. LAB: Introduction to NTP

Directions

Your instructor will assign you to work with a team of your classmates to configure an NTP server, and one or more NTP clients. Record the host names and chosen roles of your teammates' machines below.

```
NTP server: _____________

NTP client: _____________

NTP client: _____________
```

Record the commands you use to complete the steps below, and answer all questions.

Part 1: Configuring an NTP Server

The steps below should only be configured on the host you have chosen to be the NTP server. Do not start configuring the NTP clients until the server configuration is complete.

Since you probably do not have access to a radio clock in the classroom, use the NTP server's internal system clock as the authoritative time source for your team.

1. Set the local clock forward 2 minutes so the clients can actually see a clock "step" after enabling NTP.

   ```
   date MMDDhhmm
   xclock -update 1 &
   ```

2. Add a server line to the end of the **/etc/ntp.conf** file defining the local clock as the only time source. Since the internal system clock is not likely to be accurate, set the stratum level of this time source to 10.

   ```
   # vi /etc/ntp.conf
         server 127.127.1.1
         fudge 127.127.1.1 stratum 10
   ```

3. Modify the **/etc/rc.config.d/netdaemons** file to enable XNTPD on the server. Do not specify an NTP date server.

   ```
   # vi /etc/rc.config.d/netdaemons
         NTPDATE_SERVER=
         XNTPD=1
         XNTPD_ARGS=
   ```


© 2003 Hewlett-Packard Development Company, L.P.

4. Run the NTP startup script to start the **xntpd** daemon.

   ```
   # /sbin/init.d/xntpd start
   ```

5. After **xntpd** starts, it takes 5 poll cycles (320 seconds) to establish the appropriate peer and server relationships. Wait 5 minutes before proceeding on to the next question.

6. Is the **xntpd** daemon running? Are there any NTP errors in the syslog?

   ```
   # ps -e | grep xntpd

   # tail /var/adm/syslog/syslog.log
   ```

   If all is well, the daemon should be running, and there should not be any XNTPD "ERROR"s in the **syslog**.

7. Does **ntpq -p** suggest that the correct association has been formed? What stratum level did NTP assign to your local clock?

   ```
   # ntpq -p
   ```

   There should be one line in the **ntpq -p** output showing that the local clock is being used as a stratum 10 time source.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring an NTP Client

Do not start this procedure until you confirm that your NTP server is fully functional. The steps below should only be performed on the host(s) you have chosen as NTP clients.

1. Add appropriate server and drift file lines to your **/etc/ntp.conf** file to poll the NTP server created in the previous portion of the exercise.

   ```
   vi /etc/ntp.conf
   server 128.1.1.1          # assume 128.1.1.1 is the NTP srvr IP
   driftfile /etc/ntp.drift
   ```

   You may use the server's hostname rather than the IP if you wish.
   Note: **xntp** must be able to write to the directory where the drift file is located.

2. Modify the **/etc/rc.config.d/netdaemons** file to enable **xntpd**. Also, define your NTP server to be the NTPDATE_SERVER.

   ```
   # vi /etc/rc.config.d/netdaemons
         NTPDATE_SERVER=128.1.1.1   # Assume 128.1.1.1 is the NTP srvr IP
         XNTPD=1
         XNTPD_ARGS=
   ```

   Here again, you may use the server's host name in place of the IP if you wish.

3. Run the NTP startup script on the client to start the NTP daemon. Note the output as **ntpdate** steps the system clock.

   ```
   # /sbin/init.d/xntpd start
   ```

4. Check to ensure that your client formed the proper association by running **ntpq -p**.

   ```
   # ntpq -p
   ```


© 2003 Hewlett-Packard Development Company, L.P.

5. Compare the time on your client against the time on the NTP server. Do they appear to be synchronized at this point?

© 2003 Hewlett-Packard Development Company, L.P.

Module 16 — Configuring an SD-UX Server

Objectives

Upon completion of this module, you will be able to do the following:

- Create an SD-UX server.
- Copy software to an SD-UX server.
- Register an SD-UX server.
- Audit depot usage.

© 2003 Hewlett-Packard Development Company, L.P.

16–1. SLIDE: Why Create an SD-UX Depot Server?

Why Create an SD-UX Depot Server?

A network SD-UX depot server allows:

- clients without CD-ROM drives to install from the network
- software from multiple sources to be consolidated in a single location
- simplified patch and software management

Good candidates for distribution via a network depot include:

- patches
- 10.x application and core OS software
- 11.x application and core OS software
- application software

Student Notes

Most software products from HP and other vendors are shipped on CD (or DAT tape) in a Software Distributer (SD) format. Leaving the software on this distribution media has the following disadvantages:

- Systems without CD-ROM or DAT tape drives cannot install the software.
- Systems cannot easily be installed in parallel. Often, only the local system has direct access to the media.
- When installing from multiple pieces of media, the system administrator has to manage the different media (i.e. no way to consolidate into one media).
- The management of patches (which are often distributed on DAT tape) becomes increasingly complex as patch tapes accumulate.

All of the above problems can be resolved with the creation of network depots.

© 2003 Hewlett-Packard Development Company, L.P.

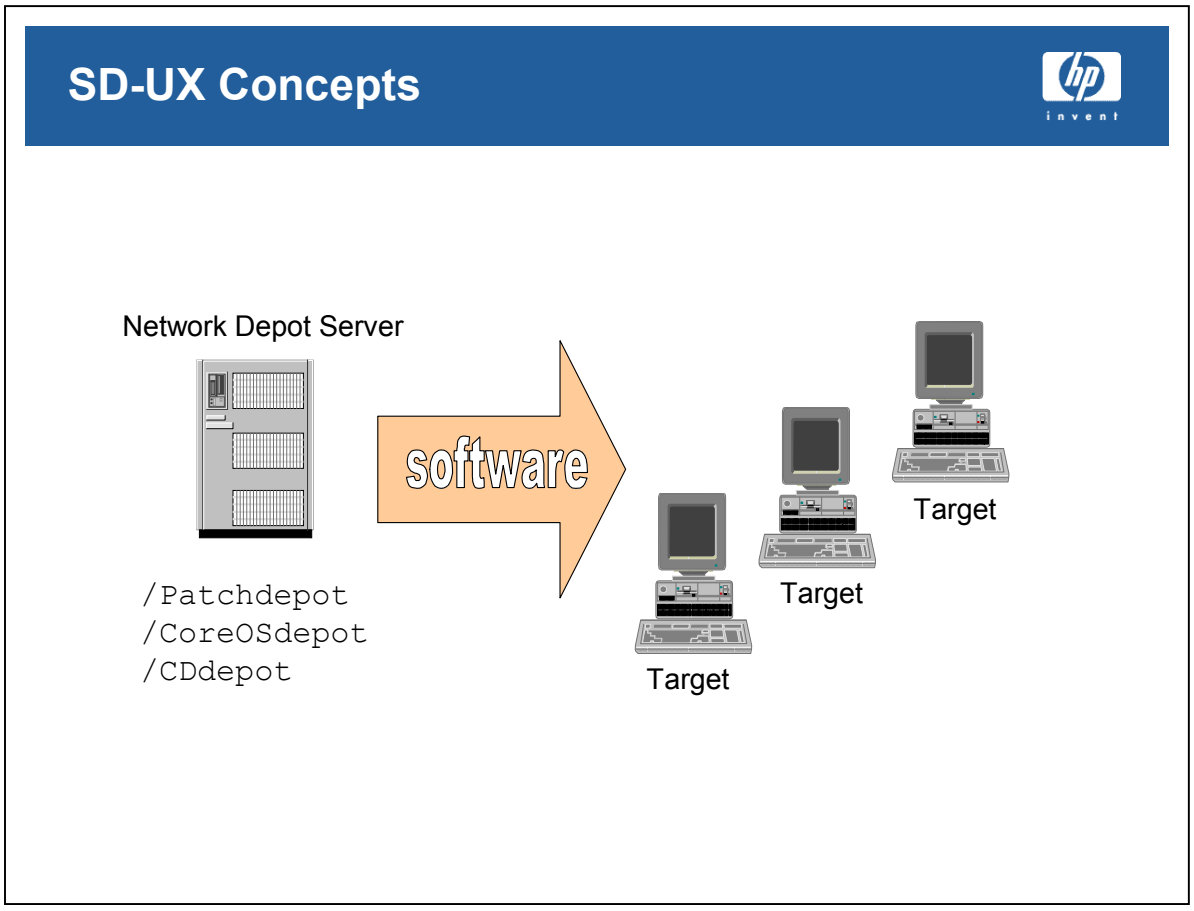
Network Depots

A network depot server is a system on the network, which contains sufficient disk space for an image of the software from CD or DAT to be off-loaded and stored to disk within the system.

The main benefit of creating a network depot on a server is that system becomes a software source for all other systems attached to the same network. Other systems on the same network can install software products from that server instead of installing them from CD-ROM or tape.

© 2003 Hewlett-Packard Development Company, L.P.

16–2. SLIDE: SD-UX Concepts



Student Notes

There are many benefits in creating network depots on servers. Three major benefits are:

Parallel installations	With a network depot, multiple systems can install software in parallel over the network. No CDs or tapes have to be carried from system to system.
Consolidated software	Software from many different sources can be consolidated into a single network depot source. This gives the system administrator valuable flexibility in organizing patches, application products, and other software products.
Better performance	Typically, installation of software from DAT tapes or CDs can be slow. Because network depots are usually stored on SCSI drives accessible over fast networks (10Mb or faster), performance is often better.

These are the main advantages of network depots, but there are others that may be of importance to certain customers, such as having the software readily available at all times. The remainder of this module will discuss how to create and administer network depots.

© 2003 Hewlett-Packard Development Company, L.P.

16–3. SLIDE: Managing Depots

Managing Depots

Registering & unregistering depots for client access

```
swreg -l  depot @ /cdrom
swreg -ul depot @ /cdrom
```

Copying software to a depot

```
swcopy
swcopy -s /cdrom perl @ /mydepot
swcopy -s /cdrom  '*' @ /mydepot
```

Removing software from a depot

```
swremove -d
swremove -d perl @ /mydepot
swremove -d  '*' @ /mydepot
```

Student Notes

OS software, patches, and applications from HP are typically provided to customers via SDUX formatted CD-ROMs or DAT tapes. Leaving the software on this distribution media has the following disadvantages:

- Systems without CD-ROM or DAT tape drives cannot install the software.
- Systems cannot easily be installed in parallel. Often, only the local system has direct access to the media.
- When installing from multiple pieces of media, the system administrator has to manage the different media (that is, no way to consolidate into one media).
- The management of patches (which are often distributed on DAT tape) becomes increasingly complex as patch tapes accumulate.

All of the above problems can be resolved with the creation of network depots.

© 2003 Hewlett-Packard Development Company, L.P.

Copying Software to a Depot

If you anticipate that many clients may install software from your depot, or if you wish to consolidate software from multiple sources into a single depot, then it may be preferable to copy software from tape and/or CD to a local depot directory. This is accomplished via the **swcopy** command.

```
# swcopy                                  (use the GUI/TUI swcopy interface)
# swcopy -s /cdrom perl @ /mydepot        (copy "perl" from CD to the /mydepot depot)
# swcopy -s /cdrom  '*' @ /mydepot        (copy all software from CD to /mydepot)
```

The **swcopy** command copies the selected software to the target, and ensures that the depot is properly registered.

Removing Software from a Depot

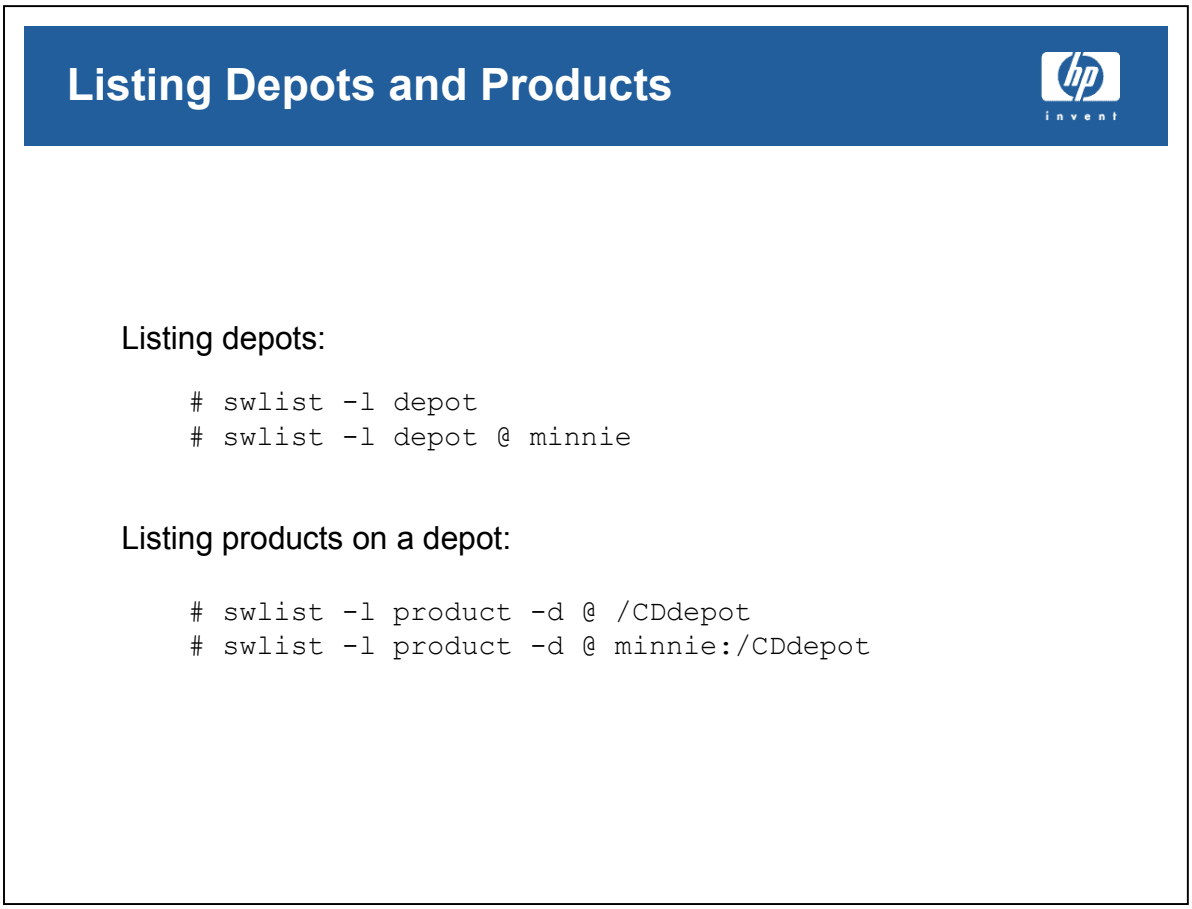
You may also remove software from an existing depot with **swremove**.

```
# swremove -d                             (remove depot software interactively)
# swremove -d perl @ /mydepot             (remove "perl" from the /mydepot depot)
# swremove -d  '*' @ /mydepot             (remove all products from /mydepot)
```

Once the last product has been removed from a depot, **swremove** automatically unregisters the depot itself.

© 2003 Hewlett-Packard Development Company, L.P.

16–4. SLIDE: Listing Depots and Products



Student Notes

When working in an environment containing network depot servers, the two basic operations performed most often are:

- Listing the names of existing depots.
- Displaying the contents (i.e. software products) contained within a specific depot.

The **`swlist`** command is used for both of these operations.

Listing the Names of Existing Depots

The command to list the names of existing depots on the local system is:

```
swlist -l depot
```

Example

```
# swlist -l depot
# Initializing...
# Target "mickie" has the following depot(s):
  /disk/diagnostics700_10.10
```


© 2003 Hewlett-Packard Development Company, L.P.

```
  /home/roc/depot_11_core
  /home/roc/glance_11_depot
  /var/spool/sw
```

Depots residing on other systems within the network can be displayed by specifying the name of the system as an argument to the **swlist** command.

```
swlist -l depot @ system_name
```

Example

```
# swlist -l depot @ minnie
# Initializing...
# Target "minnie" has the following depot(s):
  /SD_CDROM
  /var/depots/OV/mgd_node
  /patches/depot
```

Listing Software Products in Depots

The command to list the software products residing in a specific depot on the local system is

```
swlist -l product -d @ /depot_name
```

Example

```
# swlist -l product -d @ /var/spool/sw
# Initializing...
# Contacting target "mickie"...
#
# Target: mickie:/var/spool/sw
#
  Ignite-UX    B.1.47         HP-UX System Installation Services
```

The contents of a specific depot residing on another system within the network can be displayed by specifying the systems name and the depot name as arguments to the **swlist** command:

```
swlist -l product -d @ system_name:/depot_name
```

Example

```
# swlist -l product -d @ minnie:/patches/depot
# Initializing...
# Contacting target "minnie"...
#
# Target:  minnie:/patches/depot
#
  PHCO_14600    1.0            patch for lif commands
  PHCO_14777    1.0            patch for reboot.
  PHCO_14859    1.0            cumulative 10.20 libc compatibility support
  PHCO_14887    1.0            cumulative SAM/ObAM patch
  PHCO_15037    1.0            fsck_vxfs(1M) cumulative patch
  PHCO_15087    1.0            LVM commands cumulative patch
  PHCO_15217    1.0            Cumulative SCCS(1) including Year 2000 Fix
  PHCO_15219    1.0            nroff(1) Year 2000 fix
  PHCO_15220    1.0            sar(1M) Year 2000 fix
  PHCO_15221    1.0            patch with Year 2K fix
```


© 2003 Hewlett-Packard Development Company, L.P.

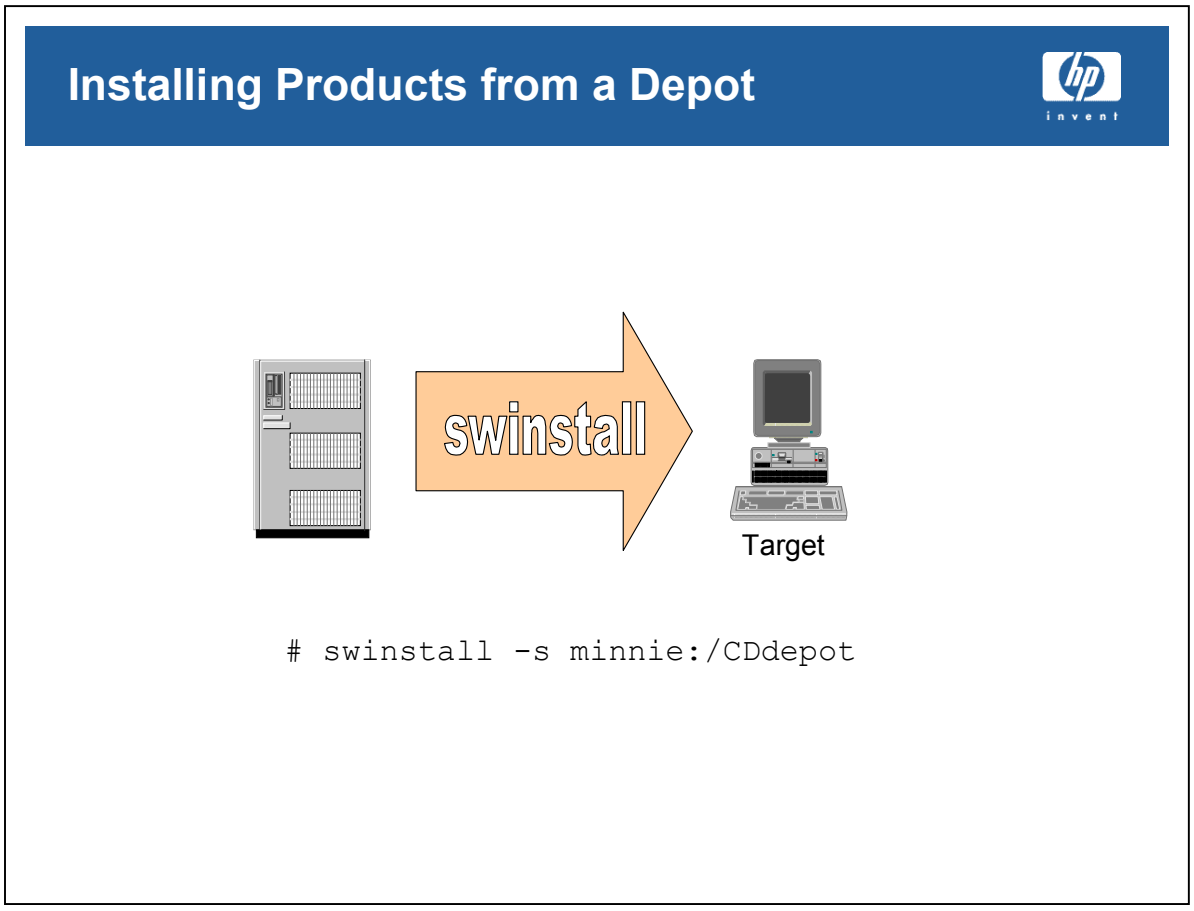
```
  PHKL_14769    1.0           11.00 data page fault panic in adjust_flock
  PHNE_12957    1.0           Bind 4.9.7 components
  PHNE_14479    1.0           ftpd(1M) and ftp(1) patch
  PHNE_14818    1.0           patch for kernel portion of Telnet
  PHNE_14819    1.0           telnetd(1M) patch
  PHNE_15047    1.0           cumulative ARPA Transport patch
```

A common error when listing depots which are CD-based, like the **/SD_CDROM** depot on system minnie (as shown in the above output), is for the CD not to be present or on-line. When this happens, the following error message will be returned when trying to access the depot:

```
# swlist -l product -d @ minnie:/SD_CDROM
# Initializing...
# Contacting target "minnie"...
ERROR:   There is currently no depot software on host "minnie" at
         location "/SD_CDROM".
ERROR:   More information may be found in the daemon logfile on this
         target (default location is minnie:/var/adm/sw/swagentd.log).
```


© 2003 Hewlett-Packard Development Company, L.P.

16–5. SLIDE: Installing Products from a Depot



Student Notes

The major benefit in creating a disk-based, network depot is all clients on the same network will be able to install software from this depot.

To access the network depot, the clients change their source location to specify the name of the server containing the depot and the name of the depot itself.

For example, if a client wanted to install the software product **`Glance`** from the depot **`/CDdepot`** on the system **`minnie`**, the command would be:

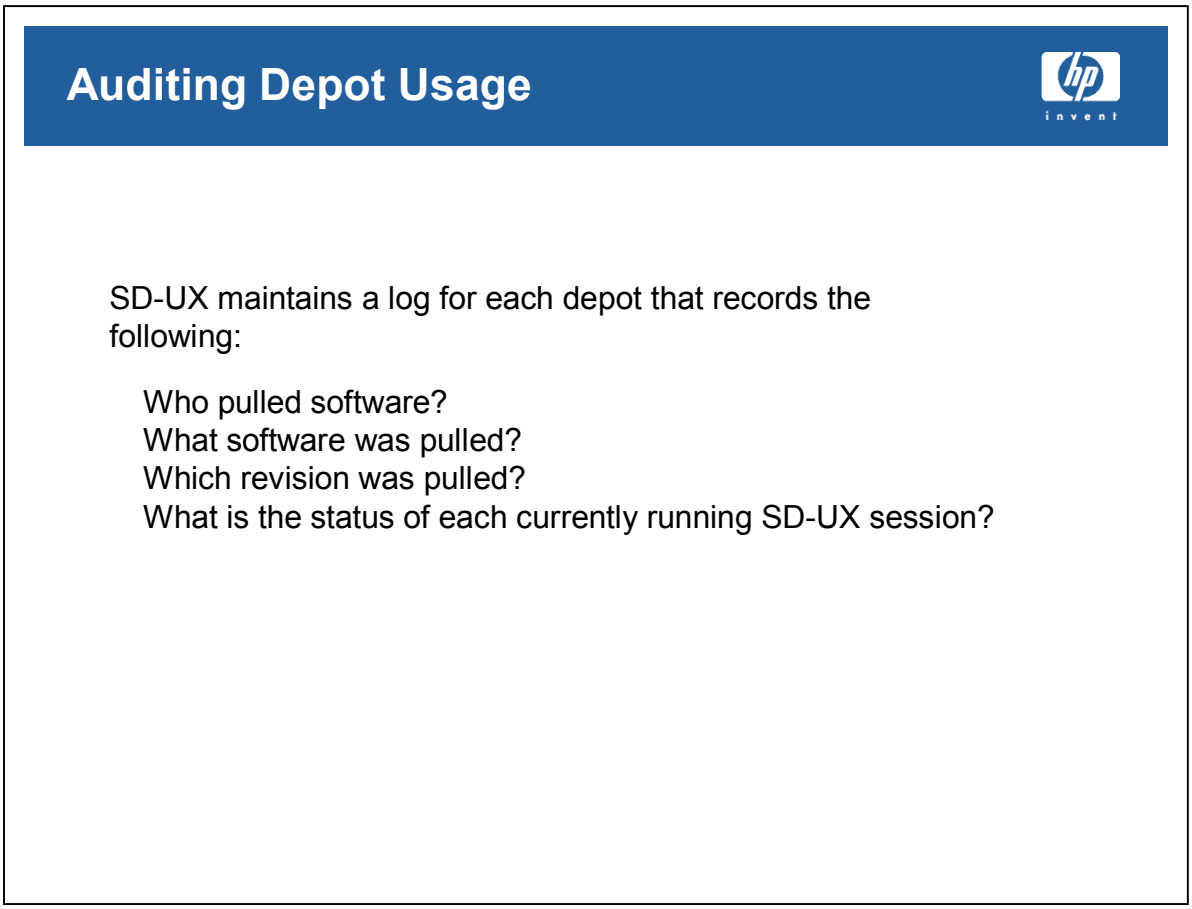
```
swinstall -s minnie:/CDdepot Glance
```

NOTE: The software can also be installed by going through the **`swinstall`** user interface.

At the screen that prompts for **Source Depot Location**, the system name **`minnie`** would be specified; at the prompt for **Source Depot Name**, the depot **`/CDdepot`** would be specified.

© 2003 Hewlett-Packard Development Company, L.P.

16–6. SLIDE: Auditing Depot Usage



Student Notes

The network depot auditing feature was introduced with the HP-UX 11.00.

Depot auditing allows the system administrator to monitor how depots within the local system are being used. Depot auditing information includes:

- What software has been copied or installed from a local depot.
- Which systems have been accessing a local depot.
- The version of the software pulled from a local depot.
- The status of the installation or copy tasks performed against a local depot.

Depot auditing information is kept for each depot in a separate log file. The log file created for each depot is called **`swaudit.log`** and is kept within each depot directory. For CD-ROM depots, the depot auditing information is kept in the **`/var/tmp`** directory.

© 2003 Hewlett-Packard Development Company, L.P.

By default, depot auditing is turned off for all depots to conserve disk space. To enable depot auditing for all depots on a system, the **swagentd** daemon needs to be restarted. The procedure for doing this is:

1. Edit the **/var/adm/sw/defaults** file and add the following line:

   ```
   swagent.source_depot_audit=true
   ```

2. Restart the **swagentd** process. If a syntax error existed in the **/var/adm/sw/defaults** file, it will be detected here.

   ```
   swagentd -r
   ```

Viewing the Audit Logs

To view the audit log file for a particular depot, change to the directory for the depot and display the **swaudit.log** file. The file may not exist if no activity has occurred against the depot.

Example

1. Perform some activity against the depot: **swlist -l depot**
2. Go to the depot directory: **cd /depot_directory**
3. View the audit log file: **more swaudit.log**

© 2003 Hewlett-Packard Development Company, L.P.

16–7. LAB: Creating and Managing an SD-UX Depot

Directions

Create a software depot and install software products from this new depot.

Perform all the tasks suggested below. Record the commands you use, and answer all questions.

1. What is an SD depot?

2. Create an SD-UX depot server, choose any path you like for your directory depot. Load only one product.

3. List the contents of your depot.

4. List a remote depot. Ask for a depot path your neighbor team and look at the manual page to get the syntax.

5. Install a product, using your own depot or a remote depot.

6. Check the installation of the product with **`swlist`** and **`swverify`**.

© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

Appendix A — Decimal-Hexadecimal-Binary Conversion

dec	hex	binary
0	00	00000000
1	01	00000001
2	02	00000010
3	03	00000011
4	04	00000100
5	05	00000101
6	06	00000110
7	07	00000111
8	08	00001000
9	09	00001001
10	0a	00001010
11	0b	00001011
12	0c	00001100
13	0d	00001101
14	0e	00001110
15	0f	00001111
16	10	00010000
17	11	00010001
18	12	00010010
19	13	00010011
20	14	00010100
21	15	00010101
22	16	00010110
23	17	00010111
24	18	00011000
25	19	00011001
26	1a	00011010
27	1b	00011011
28	1c	00011100
29	1d	00011101
30	1e	00011110
31	1f	00011111
32	20	00100000
33	21	00100001
34	22	00100010
35	23	00100011
36	24	00100100
37	25	00100101
38	26	00100110
39	27	00100111

© 2003 Hewlett-Packard Development Company, L.P.

40	28	00101000
41	29	00101001
42	2a	00101010
43	2b	00101011
44	2c	00101100
45	2d	00101101
46	2e	00101110
47	2f	00101111
48	30	00110000
49	31	00110001
50	32	00110010
51	33	00110011
52	34	00110100
53	35	00110101
54	36	00110110
55	37	00110111
56	38	00111000
57	39	00111001
58	3a	00111010
59	3b	00111011
60	3c	00111100
61	3d	00111101
62	3e	00111110
63	3f	00111111
64	40	01000000
65	41	01000001
66	42	01000010
67	43	01000011
68	44	01000100
69	45	01000101
70	46	01000110
71	47	01000111
72	48	01001000
73	49	01001001
74	4a	01001010
75	4b	01001011
76	4c	01001100
77	4d	01001101
78	4e	01001110
79	4f	01001111
80	50	01010000
81	51	01010001
82	52	01010010
83	53	01010011
84	54	01010100
85	55	01010101
86	56	01010110

© 2003 Hewlett-Packard Development Company, L.P.

87	57	01010111
88	58	01011000
89	59	01011001
90	5a	01011010
91	5b	01011011
92	5c	01011100
93	5d	01011101
94	5e	01011110
95	5f	01011111
96	60	01100000
97	61	01100001
98	62	01100010
99	63	01100011
100	64	01100100
101	65	01100101
102	66	01100110
103	67	01100111
104	68	01101000
105	69	01101001
106	6a	01101010
107	6b	01101011
108	6c	01101100
109	6d	01101101
110	6e	01101110
111	6f	01101111
112	70	01110000
113	71	01110001
114	72	01110010
115	73	01110011
116	74	01110100
117	75	01110101
118	76	01110110
119	77	01110111
120	78	01111000
121	79	01111001
122	7a	01111010
123	7b	01111011
124	7c	01111100
125	7d	01111101
126	7e	01111110
127	7f	01111111
128	80	10000000
129	81	10000001
130	82	10000010
131	83	10000011
132	84	10000100

© 2003 Hewlett-Packard Development Company, L.P.

133	85	10000101
134	86	10000110
135	87	10000111
136	88	10001000
137	89	10001001
138	8a	10001010
139	8b	10001011
140	8c	10001100
141	8d	10001101
142	8e	10001110
143	8f	10001111
144	90	10010000
145	91	10010001
146	92	10010010
147	93	10010011
148	94	10010100
149	95	10010101
150	96	10010110
151	97	10010111
152	98	10011000
153	99	10011001
154	9a	10011010
155	9b	10011011
156	9c	10011100
157	9d	10011101
158	9e	10011110
159	9f	10011111
160	a0	10100000
161	a1	10100001
162	a2	10100010
163	a3	10100011
164	a4	10100100
165	a5	10100101
166	a6	10100110
167	a7	10100111
168	a8	10101000
169	a9	10101001
170	aa	10101010
171	ab	10101011
172	ac	10101100
173	ad	10101101
174	ae	10101110
175	af	10101111
176	b0	10110000
177	b1	10110001
178	b2	10110010

© 2003 Hewlett-Packard Development Company, L.P.

```
179  b3 10110011
180  b4 10110100
181  b5 10110101
182  b6 10110110
183  b7 10110111

184  b8 10111000
185  b9 10111001
186  ba 10111010
187  bb 10111011
188  bc 10111100
189  bd 10111101
190  be 10111110
191  bf 10111111

192  c0 11000000
193  c1 11000001
194  c2 11000010
195  c3 11000011
196  c4 11000100
197  c5 11000101
198  c6 11000110
199  c7 11000111

200  c8 11001000
201  c9 11001001
202  ca 11001010
203  cb 11001011
204  cc 11001100
205  cd 11001101
206  ce 11001110
207  cf 11001111

208  d0 11010000
209  d1 11010001
210  d2 11010010
211  d3 11010011
212  d4 11010100
213  d5 11010101
214  d6 11010110
215  d7 11010111

216  d8 11011000
217  d9 11011001
218  da 11011010
219  db 11011011
220  dc 11011100
221  dd 11011101
222  de 11011110
223  df 11011111

224  e0 11100000
```


© 2003 Hewlett-Packard Development Company, L.P.

```
225  e1 11100001
226  e2 11100010
227  e3 11100011
228  e4 11100100
229  e5 11100101
230  e6 11100110
231  e7 11100111

232  e8 11101000
233  e9 11101001
234  ea 11101010
235  eb 11101011
236  ec 11101100
237  ed 11101101
238  ee 11101110
239  ef 11101111

240  f0 11110000
241  f1 11110001
242  f2 11110010
243  f3 11110011
244  f4 11110100
245  f5 11110101
246  f6 11110110
247  f7 11110111

248  f8 11111000
249  f9 11111001
250  fa 11111010
251  fb 11111011
252  fc 11111100
253  fd 11111101
254  fe 11111110
255  ff 11111111
```


© 2003 Hewlett-Packard Development Company, L.P.

Appendix B — HP-UX Administration Command Quick Reference

Modules 1- 6: TCP/IP Configuration, Routing, Troubleshooting

arp	Displays ARP cache entries.
hostname	Displays or modifies the system host name.
ifconfig	Displays or configures a network interface card.
ioscan	Scans hardware for interface cards and devices.
lanadmin	Displays or modifies a NIC card's link layer parameters.
lanscan	Lists installed LAN cards.
linkloop	Verifies link level connectivity with a test frame.
ndd	Displays or sets tunable network parameters.
netstat	Displays network interface, routing, and socket connection information.
nslookup	Tests host name resolution.
ping	Verifies network layer connectivity.
route	Adds and removes route table entries.
uname	Displays or sets the system host name.
/etc/hosts	Maps host names to IP addresses.
/etc/rc.config.d/hp*	Link layer startup script configuration files.
/etc/rc.config.d/netconf	Startup configuration file defining a host's host name and IP.
/etc/rc.config.d/nddconf	Startup configuration file defining **ndd** parameters.
/sbin/init.d/hostname	Startup script that sets the system host name.
/sbin/init.d/net	Startup script that configures LAN interface cards.

Module 7: Starting Network Services

init	Daemon responsible for managing system startup.
rc	Executes **/sbin/init.d/*** scripts to start and stop services when changing run levels.
/etc/inittab	Configuration file for the **init** daemon.
/etc/rc.config.d/*	Configuration files for **/sbin/init.d/*** scripts.
/etc/rc.log	Log file used by **/sbin/rc**.
/sbin/init.d/*	Startup scripts called by **/sbin/rc**.
/sbin/init.d/template	Template for new **/sbin/init.d/*** startup scripts.
/sbin/rc[0-4].d/	Directories consulted by **/sbin/rc** to determine which services start at which run levels.

Modules 8 - 10: NFS and AutoFS

autofs_proc	Daemon responsible for identifying idle AutoFS file systems.
automount	Used to update the mount table after modifying AutoFS maps.
automountd	AutoFS deamon that mounts and unmounts NFS file systems.
biod	Daemon that provides buffer cache functionality for NFS file systems.
exportfs	Exports and unexports directories to NFS clients.
fusers	Lists or kills processes using a mounted file system.
mount	Mounts a file system.
nfsd	NFS server daemon responsible for handling clients' access requests.
nfsstat	Displays NFS usage statistics.

© 2003 Hewlett-Packard Development Company, L.P.

portmap	Passes incoming NFS RPC requests to the appropriate RPC daemons (10.x).
rpcbind	Passes incoming NFS RPC requests to the appropriate RPC daemons (11.x).
rpcinfo	Display RPC programs registered with a host's **portmap**/**rpcbind** daemon.
rpc.lockd	Works with **rpc.statd** to provide NFS file locking.
rpc.mountd	Answers NFS mount requests.
rpc.pcnfsd	Authenticates NFS access requests from PC clients.
rpc.statd	Works with **rpc.lockd** to provide NFS file locking.
showmount	Queries an NFS server's mount daemon.
umount	Unmounts a file system.
umountall	Unmounts all file systems.
/etc/auto_master	The AutoFS master map configuration file.
/etc/auto.*	Additional AutoFS map configuration files.
/etc/exports	Lists directories to export to NFS clients at system startup.
/etc/fstab	Lists file systems to mount at system startup.
/etc/rc.config.d/nfsconf	The NFS startup configuration file.
/sbin/init.d/nfs.client	Starts NFS client functionality at system startup.
/sbin/init.d/nfs.core	Starts core NFS functionality during system startup.
/sbin/init.d/nfs.server	Starts NFS server functionality at system startup.
/var/adm/automount.log	AutoFS log file.

Module 11: NIS

domainname	Sets or displays the NIS domain name.
keyserv	Stores private encryption keys for use by secure RPCs.
nsquery	Tests user and host name lookup functionality (11.x).
passwd	Changes a password in **/etc/passwd** or the NIS passwd map.
portmap	Passes incoming NIS RPC requests to the appropriate RPC daemons (10.x).
rpcbind	Passes incoming NIS RPC requests to the appropriate RPC daemons (11.x).
rpc.yppasswdd	NIS server daemon responsible for updating user passwords in the NIS passwd map.
ypbind	NIS client daemon responsible for choosing and binding to an NIS server.
ypcat	Displays the contents of an NIS map.
ypinit	Creates NIS map databases.
ypmake	Creates or rebuilds NIS maps.
ypmatch	Searches for information in NIS maps.
yppasswd	Changes a password in an NIS password.
yppoll	Checks the status of an NIS map on a specified NIS server.
yppush	Pushes an NIS map update out to NIS slave servers.
ypserv	Daemon that answers NIS clients map lookup requests.
ypset	Binds an NIS client to a specified NIS server.
ypwhich	Displays the name of an NIS client's current NIS server.
ypxfr	Pulls a map update from an NIS master server.
ypxfrd	Transfers NIS maps between NIS master & slave servers.
/etc/nsswitch.conf	Determines what source is used for username, host name, and other lookup requests.
/etc/rc.config.d/namesvrs	NIS startup configuration file.
/var/yp/$(domainname)	Directory containing the NIS maps.

© 2003 Hewlett-Packard Development Company, L.P.

Module 12: DNS

hosts_to_named	Translates **/etc/hosts** into DNS database files.
named	DNS server daemon.
nslookup	Interactively query and troubleshoot DNS servers.
nsquery	Query and troubleshoot the local host name resolver.
sig_named	Send restart and other signals to the DNS named daemon.
/etc/named.boot	DNS named daemon's boot configuration file.
/etc/named.data/*	DNS database files.
/etc/nsswitch.conf	Determines what source is used for host name and other lookup requests.
/etc/resolv.conf	Resolver configuration file.
/sbin/init.d/named	DNS **named** daemon startup script.
/var/adm/syslog/syslog.log	Log file used by **named**, and many other daemons and services.

Module 13: Configuring and Securing ARPA/Berkeley Services

inetd	Superdaemon responsible for invoking internet service server processes as needed.
telnetd	**telnet** server process.
ftpd	**ftp** server process.
netstat	Displays, among other things, socket connections.
remshd	Remote shell server process.
rlogind	**rlogin** server process.
/etc/ftpd/ftpusers	Defines which users may access a host via FTP.
/etc/hosts.equiv	Grants password free Berkeley Service access to selected clients.
/etc/inetd.conf	Determines which internet services **inetd** should and shouldn't provide.
/etc/services	Maps port numbers to service names.
/var/adm/inetd.sec	Determines which clients can access which **inetd** services.
~/.rhosts	Grants password free Berkeley Service access to selected clients.
~/.netrc	Contains login information by the **ftp** autologin process.

Module 14: BOOTP/TFTP

bootpd	Provides IP configuration information for BOOTP and DHCP clients.
tftpd	Provides password-free FTP-like access to allow network printers and other devices to download configuration files.
jetadmin	Menu-based utility for configuring network printer BOOTP/TFTP service.
xtadm	Menu-based utility for configuring X-terminal BOOTP/TFTP service.
/etc/bootptab	The BOOTP configuration file.
/etc/inetd.conf	The **inetd** configuration file, used to enable/disable BOOTP/TFTP service.
/etc/services	Determines which port numbers are used by **bootp** and **tftp**.
/home/tftpdir/	TFTP home directory: the only directory normally accessible to TFTP clients.

© 2003 Hewlett-Packard Development Company, L.P.

Module 15: NTP

ntpdate	Polls one or more NTP servers and immediately, and adjusts the system clock accordingly.
ntpq	Displays NTP status information.
xntpd	Polls one or more NTP servers at regular intervals, adjusting the system clock as necessary.
/etc/ntp.drift	File used by **xntpd** to track the accuracy of the system clock over time.
/etc/ntp.conf	The **xntpd** configuration file.
/etc/rc.config.d/netdaemons	Startup script configuration file for NTP and other services.
/sbin/init.d/xntpd	The NTP startup script.

Module 16: Configuring an SD-UX Server

swagent	Process responsible for installing, removing, and copying SD-UX software.
swagentd	Daemon responsible for starting **swagent** processes as necessary.
swcopy	Copies SD-UX software between depots.
swinstall	Installs SD-UX software from a depot.
swlist	Lists SD-UX software in a depot, or installed on a host.
swreg	Registers or unregister an SD-UX depot.
swremove	Removes SD-UX software from a depot or host.

© 2003 Hewlett-Packard Development Company, L.P.

Solutions

© 2003 Hewlett-Packard Development Company, L.P.

© 2003 Hewlett-Packard Development Company, L.P.

1–23. REVIEW QUESTIONS: LAN Concepts and Components

Directions

Answer the following questions:

1. If a host has two LAN interface cards, will the MAC addresses of the two cards be the same, or different?

   Answer

   Different. Every LAN card should have a unique MAC address.

2. Is it possible to determine which network a host is on just by looking at the host's MAC address?

   Answer

   No. Given a host's IP address and netmask you can determine which network the host is on, but a MAC address alone is insufficient.

3. Complete the following table:

IP Address	Netmask	Network Address	Broadcast Address
167.12.132.5/16			
124.132.12.5/8			
213.1.231.45/24			

   Answer

IP Address	Netmask	Network Address	Broadcast Address
167.12.132.5/16	255.255.0.0	167.12.0.0/16	167.12.255.255
124.132.12.5/8	255.0.0.0	124.0.0.0/8	124.255.255.255
213.1.231.45/24	255.255.255.0	213.1.231.0/24	213.1.231.255

4. Which of the networks listed in question 3 would allow the fewest hosts? What is the maximum number of hosts allows on that network?

   Answer

   The 213.1.231.0/24 network has the fewest host bits, so it would support the fewest hosts. With 8 host bits, this network could have at most $2^8 = 256$ addresses. Subtracting the broadcast and network addresses means that the network could support no more than 254 hosts.

5. How many different networks are represented by the list of IP addresses below?

```
132.1.1.3/16
132.2.1.1/16
132.1.1.2/16
132.1.1.1/16
132.1.2.1/16
132.1.2.2/16
```


© 2003 Hewlett-Packard Development Company, L.P.

Answer

The /16 tells us that there are 16 network bits in each of these IP addresses. Thus, the first two octets define the network portion of the IP. This suggests that just two networks are represented in this list: 132.1.0.0/16 and 132.2.0.0/16.

6. What is the highest possible host IP address on the 158.153.0.0/16 network? What is the lowest possible host IP address on this network?

Answer

The highest host IP is 158.153.255.254.
The lowest host IP is 158.153.0.1.

7. What is the difference between a destination port number and a destination IP address?

Answer

A destination IP determines which host should receive a packet. A destination port number determines which application on a host should receive a packet.

8. Name one major difference between UDP and TCP.

Answer

TCP is a connection-oriented protocol that provides a built-in acknowledgement mechanism. UDP is a connection-less protocol that does not provide an acknowledgement mechanism.

9. HPUX provides three different methods for mapping host names to IP addresses. Name two.

Answer

`/etc/hosts`, DNS, and NIS may all be used to resolve host names to IP addresses.

© 2003 Hewlett-Packard Development Company, L.P.

3–9. LAB: Configuring Network Connectivity

Directions

This lab will configure a new host name and IP address for each system in your classroom. Please check with the instructor for an assigned host name and IP address.

Preliminary Step

Just in case something goes wrong during this lab, make a backup copy of all of your network configuration files. There is a shell script in your **`labs`** directory designed specifically for this purpose. The shell script will save a tar archive backup of your network configuration files in the file you specify. Run the shell script by typing:

```
# /labs/netfiles.sh -s ORIGINAL
```

Part 1: Checking the Current LAN Card Configuration

Check the current configuration of the LAN card. Answer the following questions related to its configuration.

1. How many LAN cards does your system have, and what are their Hardware paths?

   Answer

   The following commands may be used to view your LAN card hardware paths:

   ```
   # lanscan
   # ioscan –funC lan
   ```

2. Verify that the "Networking" product is installed on your machine. Is any additional networking software installed on your machine to support your LAN interface cards?

   Answer

   ```
   # swlist –l product Networking
   ```

   Every machine should have the Networking product loaded. Other LAN software will vary from system to system.

3. Does your kernel contain the drivers necessary to support your LAN cards? Which command will tell you if a driver has CLAIMED your LAN cards? If your LAN card is UNCLAIMED, install the necessary drivers.

   Answer

   ```
   # ioscan –funC lan
   ```

   Look for "UNCLAIMED" LAN cards. The drivers should already be installed, and all cards should be "CLAIMED."

© 2003 Hewlett-Packard Development Company, L.P.

4. Do device files exist for your LAN cards?

   Answer

   ```
   # ioscan -funC lan
   ```

   The device files should already exist.

5. List the current MAC address, IP address, netmask, and broadcast address for each of your LAN cards.

   Answer

   ```
   # lanscan            shows the MAC address
   # ifconfig lan0      shows the IP, netmask, and broadcast addresses
   ```

   Note that these solutions assume that your default LAN card is lan0. The default LAN interface name on your system may be different. The IP, netmask, and broadcast addresses will also vary from classroom to classroom.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring the New LAN Card Configuration

The goal of this portion of the lab exercise is to configure a new IP address and host name for each of the machines in the classroom. Your instructor will assign you a host name/IP from the table that follows. All of the addresses listed are on the 128.1.0.0/16 network.

```
corp        128.1.0.1
sanfran     128.1.1.1
oakland     128.1.1.2
la          128.1.1.3

chicago     128.1.2.1
peoria      128.1.2.2
rockford    128.1.2.3

atlanta     128.1.3.1
athens      128.1.3.2
macon       128.1.3.3

nyc         128.1.4.1
albany      128.1.4.2
buffalo     128.1.4.3

paris       128.1.5.1
lyon        128.1.5.2
grenoble    128.1.5.3

london      128.1.6.1
leeds       128.1.6.2
ipswich     128.1.6.3

bonn        128.1.7.1
berlin      128.1.7.2
hamburg     128.1.7.3

tokyo       128.1.8.1
kyoto       128.1.8.2
osaka       128.1.8.3
```

Changing your host name and IP on a running system can wreak havoc on CDE and other applications.

1. Kill CDE before going any further:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. Is your lan0 card still "UP" after killing CDE? Look at the "flags" listed in the output from the **`ifconfig`** command.

© 2003 Hewlett-Packard Development Company, L.P.

Answer

```
# ifconfig lan0
```

The LAN card should still be up.

3. From the command line, change your IP to the address suggested by your instructor from the table above. Also set your netmask to a value appropriate for a /16 network.

Answer

```
# ifconfig lan0 128.1.1.1 netmask 255.255.0.0
```

The solutions assume that you have been assigned host name sanfran, and IP address 128.1.1.1. Your IP address and host name may be different.

4. Is your new IP address set properly? How can you find out?

Answer

```
# ifconfig lan0
```

ifconfig should indicate that the IP and netmask have been set properly.

5. Modify the appropriate startup file to make your IP address change permanent. Allow the system to default the broadcast address. Also, permanently change your host name in this startup file.

Answer

```
# vi /etc/rc.config.d/netconf
  HOSTNAME=sanfran              use your new host name here
  INTERFACE_NAME[0]=lan0        use your interface name here
  IP_ADDRESS[0]=128.1.1.1       use your new IP here
  SUBNET_MASK[0]=255.255.0.0
  BROADCAST_ADDRESS[0]=""
  INTERFACE_STATE[0]=""
  DHCP_ENABLE[0]=""
```

6. The system keeps sanitized copies of many system configuration files in **/usr/newconfig/etc**. Overwrite your current **/etc/hosts** file with a copy of **/usr/newconfig/etc/hosts**.

Answer

```
# cp /usr/newconfig/etc/hosts /etc/hosts
```

7. Add the hosts and IP addresses from the table above to your new **/etc/hosts** file.

Answer

```
# vi /etc/hosts
```


© 2003 Hewlett-Packard Development Company, L.P.

```
128.1.1.1 sanfran        add all other host/IP pairs from the table above
```

8. Define a host name alias for each of the host names in your row. Use the first name of the user sitting at each station as the alias.

Answer

```
# vi /etc/hosts
  128.1.1.1 sanfran student1
```

9. Reboot to see if your changes worked!

Answer

```
# shutdown -ry 0
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Checking the New Configuration

1. Check your LAN card's IP. Did the configuration work?

   Answer

   ```
   # ifconfig lan0
   ```

   The configuration should have succeeded!

2. The **hostname** command will display your system host name. Check to ensure that your host name is set properly.

   Answer

   ```
   # hostname
   ```

   Your host name should be set properly.

3. Based on your Answers to questions 1 and 2 above, what commands did the **/sbin/init.d/net** script appear to execute on your behalf during the boot process?

   Answer

   The system appears to have executed

   ```
   # hostname sanfran
   # ifconfig lan0 128.1.1.1 netmask 255.255.0.0 up
   ```

4. Try to **ping** the *IP address* of one of your classmates who has finished rebooting. Does this work?

   Answer

   ```
   # ping 128.1.0.1
   ```
   use your neighbor's IP address here.

   This should succeed!

5. Try to **ping** the host name of one of your classmates who has finished rebooting. Does this work?

   Answer

   ```
   # ping corp
   ```
   use your neighbor's host name here.

   Assuming the host name you ping has been added to **/etc/hosts**, and that host is configured properly, this should work.

6. Try to **ping** a neighboring machine using the alias you defined in your hosts file. Does this seem to work?

© 2003 Hewlett-Packard Development Company, L.P.

Answer

```
# ping instructor
```

This should succeed, too.

© 2003 Hewlett-Packard Development Company, L.P.

4–7. LAB: Configuring Routing

Directions

Record the commands you use to perform the tasks suggested below.

Your instructor has configured one of the nodes in your classroom network as a router with two interfaces:

Router's first LAN card IP:	128.1.0.1/16
Router's second LAN card IP:	192.1.1.1/24

Part 1: Viewing and Modifying the Routing Table

1. View your routing table. What routes are currently defined on your host?

   Answer

   ```
   # netstat –rn
   ```

   You should have routes defined to:
   - your own IP address,
   - the 128.1.0.0 network,
   - the 127.0.0.1 address, and
   - the 127.0.0.0 network.

2. Are you able to ping the first LAN card on the router?
   Are you able to ping the second LAN card on the router? Explain!

   Answer

   You should be able to **`ping`** the 128.1.0.1 address on the router since your host is on the 128.1.0.0/16 network.

   The second LAN card, however, is on a different network. Since there is not an entry in your routing table for the 192.1.1.0/24 network, you should not be able to **`ping`** 192.1.1.1.

3. From the command line, add a route to the 192.1.1.0/24 network. Then check your routing table again to verify that you were successful.

   Answer

   The 192.1.1.0/24 network is accessible via the router at address 128.1.0.1.

   ```
   # route add net 192.1.1.0 netmask 255.255.255.0 128.1.0.1 1
   # netstat -rn
   ```


© 2003 Hewlett-Packard Development Company, L.P.

4. Can you **ping** the 192.1.1.1 LAN card on **corp** now?

   Answer

   ```
   # ping 192.1.1.1
   ```

   Now that you have a route to the 192.1.1.0 network, this should succeed.

5. Delete the 192.1.1.0/24 route table entry. Then check the routing table again to verify that you were successful.

   Answer

   ```
   # route delete net 192.1.1.0 netmask 255.255.255.0 128.1.0.1
   # netstat –rn
   ```

6. Define the router that was configured by your instructor as your default router. Then check your routing table again to be sure this worked.

   Answer

   ```
   # route add default 128.1.0.1 1
   # netstat -rn
   ```

7. Can you ping 192.1.1.1 now, even though you do not have an explicit route to 192.1.1.1?

   Answer

   ```
   # ping 192.1.1.1
   ```

   This should work! Although there is not an explicitly defined route to the 192.1.1.0/24 network, the system uses the default route you just defined. Since the default router has a connection to the 192.1.1.1 network, this **ping** should succeed.

8. How can you ensure that your default route is defined after every system boot? Make it so.

   Answer

   ```
   # vi /etc/rc.config.d/netconf
      ROUTE_DESTINATION[0]=default
      ROUTE_MASK[0]=""
      ROUTE_GATEWAY[0]=128.1.0.1
      ROUTE_COUNT[0]=1
   ```

9. Reboot your machine. When your machine comes back up again, check the routing table to verify that the default route is defined.

   Answer

   ```
   # shutdown –ry 0
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Adding Router Entries to the /etc/hosts File

1. Add an entry to your **/etc/hosts** file for **corp**'s second LAN card. Since both 128.1.0.1 and 192.1.1.1 are on the same machine, both IP addresses should be mapped to host name **corp**.

```
# vi /etc/hosts
   128.1.0.1 corp
   192.1.1.1 corp
```

2. If you **ping corp**, which of corp's IP addresses does your system appear to choose? Watch your **ping** output carefully.

   Answer

```
# ping corp
```

   The system appears to ping the first address listed in **/etc/hosts**, which is 128.1.0.1 in this case.

3. For troubleshooting purposes, it may be helpful to be able to specify which IP address is used when pinging a router such as **corp**. You may wish to assign **/etc/hosts** aliases to each of the LAN cards on **corp**.

   Answer

```
# vi /etc/hosts
  128.1.0.1 corp corp128
  192.1.1.1 corp corp192
```

4. How can you specifically **ping** the 192.1.1.1 interface card on **corp** now? How can you specifically **ping** the 128.1.0.1 interface on "**corp**"?

   Answer

```
# ping corp192
# ping corp128
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Important! Backup Your New Network Configuration!

1. Use the **netfiles** script to backup the new network configuration that you configured over the last couple of chapters. Many of the labs that follow in this course require access to this archive backup!

```
# /labs/netfiles.sh -s NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

5–13. LAB: Configuring Subnets

Directions

Answer all of the questions below. Assume that your network contains some older devices that do not support all-0 or all-1 subnet addresses.

Part 1

1. Your company's network address is 128.20.0.0/16, but your netmask is set to 255.255.255.0. Given this netmask, how many bits are in the subnet portion of your IP address?

   Answer

   The /16 appended to the end of the network IP address indicates that the first 16 bits (or first two octets) contain network bits. The netmask indicates that the first three octets are all masked. Thus, all 8 bits in the third octet must be subnet bits.

2. Given your answer to the previous question, how many host addresses may be configured on each subnet?

   Answer

   With 8 bits, it is possible to represent 2^8 = 256 addresses. However, each subnet must have a subnet address and a broadcast address. Thus, each subnet could have at most 254 hosts.

3. What are the lowest and highest subnet addresses?

   Answer

   The lowest subnet address is 128.20.1.0.
   The highest subnet address is 128.20.254.0.

4. What are the lowest and highest host addresses on the first subnet?

   Answer

   The lowest host address on the first subnet is 128.20.1.1.
   The highest host address on the first subnet is 128.20.1.254.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2

Your company's network address is 192.30.40.0/24, and you need to create two subnets.

1. How many contiguous bits are needed, and in which octet?

   Answer

   Two bits are required to form two subnets.
   The /24 indicates that the first three octets are network octets.
   Thus, the subnet bits must be taken from the fourth octet.

2. What is the subnet mask?

   Answer

   We need to mask the network bits in the first three octets, as well as the two subnet bits in the fourth octet. This yields netmask value 255.255.255.192.

   ```
   255.255.255.11000000 = 255.255.255.192
   ```

3. What are the valid subnet addresses?

   Answer

   The valid subnets would be:

   ```
   192.30.40.01000000 = 192.30.40.64/26
   192.30.40.10000000 = 192.30.40.128/26
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3

Your company's network address is 132.40.0.0/16. You need to configure nine subnetworks.

1. How many bits are needed to form 9 subnets?

   Answer

   Three subnet bits yield six subnets.
   Four subnet bits yield fourteen subnets.
   To meet the stated requirements, we must use four bits.
   The extra subnets may be preserved for future growth.

2. What will be the subnet mask in dotted decimal notation?

   Answer

   The subnet mask must be:

   ```
   255.255.11110000.00000000 = 255.255.240.0
   ```

3. List the first three subnet addresses.

   Answer

   The first three subnets would be:

   ```
   132.40.00010000.00000000 = 132.40.16.0/20
   132.40.00100000.00000000 = 132.40.32.0/20
   132.40.00110000.00000000 = 132.40.48.0/20
   ```

4. How many hosts can be on each subnet?

   Answer

   Since there are 4 host bits in the third octet, and 8 host bits in the fourth octet, we have a grand total of 12 host bits. With 12 host bits, we can represent 2^{12} = 4096 addresses. Subtracting the subnet address and broadcast address, we are left with 4094 host addresses per subnet.

5. What is the complete address for the first host on the first subnet?

   Answer

   The address of the first host on the first subnet must be:

   ```
   132.40.00010000.00000001 = 132.40.16.1/20
   ```


© 2003 Hewlett-Packard Development Company, L.P.

6. What would be the complete address for the last host on the first subnet?

   Answer

   To formulate the address of the last host on the first subnet, set all but the last host bit to "1". This yields:

   ```
   132.40.00011111.11111110 = 132.40.31.254/20
   ```

7. Fill in the variable values you would expect to see in the **/etc/rc.config.d/netconf** file for the last host on the first subnet. Record the variable values below, but do not actually modify the **/etc/rc.config.d/netconf** file on your system.

   ```
   INTERFACE_NAME[0]=lan0
   IP_ADDRESS[0]=
   SUBNET_MASK[0]=
   ```

   Answer

   ```
   INTERFACE_NAME[0]=lan0
   IP_ADDRESS[0]=132.40.31.254
   SUBNET_MASK[0]=255.255.240.0
   ```

8. What command would the **/sbin/init.d/net** script execute because of the **netconf** values in the previous question?

   Answer

   ```
   ifconfig lan0 132.40.31.254 netmask 255.255.240.0 up
   ```


© 2003 Hewlett-Packard Development Company, L.P.

6–12. LAB: Troubleshooting Network Connectivity

Directions

Answer all questions below. Also, record the commands you use to find the answers.

Preliminary

Over the course of this lab, you will be asked to disable your LAN card, which can cause serious problems for CDE. Before starting the lab, shut down CDE:

```
# /sbin/init.d/dtlogin.rc stop
```

Part 1: Determining Your Current Network Configuration

1. Determine your host name, and MAC address and IP address of your lan interface(s).

   ```
   MAC address(es) :
   IP address(es)  :
   Hostname        :
   ```

   Answer

   ```
   # lanscan                    # shows your MAC address
   # ifconfig lan0              # shows your IP address
   # hostname                   # shows your host name
   ```

2. To which network are you directly connected?

   Do you have a default route defined so you can reach other networks?

   Answer

   ```
   # netstat -i                 # shows your network address
   # netstat -rn                # shows your routing table
                                  (including the default route)
   ```

3. Given a host name, how can you determine the IP address of another machine? Ask your neighbor for their host name, then determine their IP.

   Answer

   ```
   # nslookup sanfran           # Use your neighbor's host name.
   ```

4. Now that you know your neighbor's IP address, how can you determine their MAC address? Do it.

   Answer

   ```
   # ping 128.1.1.1             # ping their IP to add it to the
                                  arp cache
   @ # arp -a | grep 128.1.1.1  # now find their IP and MAC in the
                                  arp cache
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Testing LAN Connectivity

1. Ensure that your lan0 card and your neighbor's lan0 interface card are both in an "UP" state. Can you ping your neighbor's IP address?

   Answer

   ```
   # ping 128.1.1.1 #      Use your neighbor's IP address. This should succeed.
   ```

2. What happens if your LAN card is "DOWN"? Change the IP configuration state of your lan0 interface to "DOWN." What appears in **`netstat -i`** now for your card?

   Answer

   ```
   # ifconfig lan0 down
   # netstat –i
   # telnet corp
   ```

   The card now has an "*" in the first column to indicate that the card is down.

3. While your LAN card is DOWN, can you ...

   Ping your neighbor's IP address?
   Ping your own IP address?
   Ping your loopback address?

   Answer

   Ping hangs when you attempt to hit your neighbor's IP. However, you may be surprised to discover that pinging your own IP or your loopback address works.

   Accessing your own IP or the loopback address bypasses your LAN card altogether and should succeed regardless of the state of your LAN card.

4. Now try linkloop'ing to your neighbor's MAC address. Does this work? Explain.

   Answer

   `linkloop` should succeed, even though ping fails.

   `linkloop` is an OSI layer 2 utility that succeeds regardless of the IP configuration of the card.

5. Based on your answer to the previous question, when might **`linkloop`** be useful?

   Answer

   `linkloop` can test connectivity between any two hosts on a network even if the IP configuration on either host is garbled. If you can **`linkloop`** a host, but cannot **`ping`** that same host, you may want to check the routing tables and IP addresses on both machines.

6. Bring your lan0 card back to an "UP" state.

   Answer

   ```
   # ifconfig lan0 up
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Troubleshooting Connectivity Problems

1. Before starting this exercise, make sure you are able to **ping** host name "**corp**," at IP address 128.1.0.1.

2. There should be a shell script in your **/labs** directory called **/labs/corrupt.sh**. Run the script. When prompted, enter a number between 1 and 6. Based on your response, the script will corrupt your LAN configuration in one of six different ways. When the script terminates, your task is to fix your LAN configuration so the command **ping corp** succeeds. Take advantage of all the tools we discussed in this chapter.

3. Once you successfully troubleshoot and fix your configuration, run the script again, choose a different number, and again fix the resulting problem. If time permits, try each of the 6 options provided by the **corrupt.sh** script.

Good luck!

Part 4: Cleanup

Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

7–11. LAB: Starting Network Services

Directions

Work on your own to perform the following tasks.

Part 1: Exploring the Startup/Shutdown Scripts

You have seen in this chapter that many system and network services are started automatically during the boot process via "S" scripts in the **/sbin/rc*.d** directories. You can view a list of these scripts by typing:

```
# ls /sbin/rc*.d/S*
```

Answer the questions below using the output from the **ls** command above.

1. At which run level does NFS client functionality start?

   Answer

   NFS client functionality starts at run level 2.

2. At which run level does NFS server functionality start?

   Answer

   NFS server functionality starts at run level 3.

3. At which run level does your system set its host name?

   Answer

   The host name is set at run level 1.

4. At which run level does the **net** script set your IP address?

   Answer

   Run level 2.

5. At which run level does the **sendmail** daemon begin delivering mail?

   Answer

   Run level 2.

6. At which run level does the NIS service become available?

   Answer

   Run level 2.

© 2003 Hewlett-Packard Development Company, L.P.

7. At which run level does the system enable access to **`ftp`**, **`telnet`**, and other Internet services?

   HINT: Internet services are started by the **`inetd`** Internet daemon.

   Answer

   Run level 2.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Starting and Stopping Services

Most services may be manually started and stopped using the startup scripts in the **/sbin/init.d** directory.

1. Is the **sendmail** daemon currently running on your machine?

   Answer

   ```
   # ps -e | grep sendmail
   ```

   Answer=Yes (On most systems, **sendmail** should be running by default.)

2. Stop the **sendmail** daemon using the **init.d** script.

   Answer

   ```
   # /sbin/init.d/sendmail stop
   ```

3. Is the **sendmail** daemon running?

   Answer

   ```
   # ps -e | grep sendmail
   ```

   Sendmail is not running.

4. Restart **sendmail** properly, then check to ensure the daemon is running

   Answer

   ```
   # /sbin/init.d/sendmail start
   # ps -e | grep sendmail
   ```

   The daemon should be running.

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Enabling, Disabling, and Configuring Services

There are many network and system services available, but you may not need all of those services to be enabled. For instance, if you do not use networked file systems, you may choose to disable NFS. Most services may be enabled or disabled via their control variables. Usually control variables match the name of the service they control, for example, the **sendmail** daemon is controlled by the SENDMAIL control variable.

Setting a control variable to "1" enables that service at next boot, while setting the control variable to "0" disables the service at next boot. Control variables are set in configuration files in **/etc/rc.config.d/*** . Sometimes the configuration file matches the name of the service. You can always use the **grep** command to find the proper configuration file for a service. For instance, the output from the following **grep** command suggests that the sendmail control variable is defined in **/etc/rc.config.d/mailservs**.

```
# grep -il sendmail /etc/rc.config.d/*
/etc/rc.config.d/mailservs
```

See if you can find the **/etc/rc.config.d** configuration files for each of the services below, and determine which of those services are enabled on your system.

Service Name	Configuration File Name	Enabled?
nfs.server	**/etc/rc.config.d/nfsconf**	Y
nfs.client	**/etc/rc.config.d/nfsconf**	Y
nis.server	**/etc/rc.config.d/namesvrs**	N
nis.client	**/etc/rc.config.d/namesvrs**	N
sendmail	**/etc/rc.config.d/mailsvrs**	Y
named (DNS)	**/etc/rc.config.d/namesvrs**	N
xntpd	**/etc/rc.config.d/netdaemons**	N

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Creating a Custom Startup Script

In this part of the lab exercise, you will have an opportunity to create a custom startup/shutdown script to start and stop the **pfs_mountd** daemon used by the PFS file system in HP-UX. The "Portable File System" is one of the few services in HP-UX that does not include a pre-configured startup script, so this is a particularly practical exercise!

1. Make a copy of the **/sbin/init.d** to use as a template for your **pfs_mountd** startup script.

```
# cp /sbin/init.d/template /sbin/init.d/pfs_mountd
```

2. Use your editor of choice to customize the new startup script.

```
# vi /sbin/init.d/pfs_mountd
```

   a. Scroll down to the **case** statement towards the middle of the script. Look for the following:

```
'start_msg')
      # Emit a _short_ message relating to running this script
      # with the "start" argument; this message appears as part
      # of the checklist.
       echo "Starting the <specific> subsystem"
       ;;
```

   Change the **echo** statement to the following:

```
'start_msg')
    # Emit a _short_ message relating to running this script
    # with the "start" argument; this message appears as part
    # of the checklist.
    echo "Starting the pfs_mountd subsystem"
    ;;
```

   b. Scroll down to the **stop_msg** portion of the **case** statement that looks like this:

```
'stop_msg')
    # Emit a _short_ message relating to running this script
    # with the "stop" argument; this message appears as part
    # of the checklist.
    echo "Stopping the <specific> subsystem"
    ;;
```

   Change the **echo** statement to the following:

```
'stop_msg')
    # Emit a _short_ message relating to running this script
    # with the "stop" argument; this message appears as part
    # of the checklist.
    echo "Stopping the pfs_mountd subsystem"
    ;;
```


© 2003 Hewlett-Packard Development Company, L.P.

c. Scroll down to the **start** argument in the **case** statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   :
fi
;;
```

Change the **CONTROL_VARIABLE**, and add the command necessary to start **pfs_mountd** as shown below. Also add a call to the **set_return** function to notify **/sbin/rc** if the daemon successfully starts:

```
# Check to see if this script is allowed to run...
if [ "$PFS_MOUNTD" != 1 ]; then
   rval=2
else
   # Execute the commands to start your subsystem
   /usr/sbin/pfs_mountd &
   set_return
   :
fi
;;
```

d. Next, scroll down to the `stop` argument in the `case` statement that looks like this:

```
# Check to see if this script is allowed to run...
if [ "$CONTROL_VARIABLE" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
fi
;;
```

Change the **CONTROL_VARIABLE**, and add the command necessary to kill **pfs_mountd** as shown below. Also add a call to the **set_return** function to notify **/sbin/rc** if the daemon successfully starts:

```
# Check to see if this script is allowed to run...
if [ "$PFS_MOUNTD" != 1 ]; then
   rval=2
else
   :
   # Execute the commands to stop your subsystem
   kill $(ps -ef | grep /usr/sbin/pfs_mountd | grep -v grep |\
   cut -c10-14)
   set_return
```


© 2003 Hewlett-Packard Development Company, L.P.

```
    fi
    ;;
```

e. Save your changes and quit **/sbin/init.d/pfs_mountd**.

3. Create a configuration file and a control variable for your new startup script:

```
# vi /etc/rc.config.d/pfs_mountd
  PFS_MOUNTD=1
```

4. Create a start link to start the new service at run level 3 using the "don't care" 900 sequence number, and a kill link to kill the new service with sequence number 100 at run level 2:

```
# ln –s /sbin/init.d/pfs_mountd /sbin/rc3.d/S900pfs_mountd
# ln –s /sbin/init.d/pfs_mountd /sbin/rc2.d/K100pfs_mountd
```

5. Test your new startup script by executing both the **start** and **kill** links.

```
# /sbin/rc3.d/S900pfs_mountd start
# ps –ef | grep pfs_mountd
# /sbin/rc2.d/K100pfs_mountd stop
# ps –e
```

6. Assuming the previous test succeeded, change run levels a few times to further test your scripts.

```
# init 2
# init 3
# init 2
```

Note that the first **init 2** may fail. Can you explain why?

Answer

The **pfs_mountd** daemon is not running initially, so the **kill** command fails. Bouncing back up to run level 2 starts the daemon, so the second attempt to run the start script succeeds.

© 2003 Hewlett-Packard Development Company, L.P.

9–19. LAB: Configuring NFS

Directions

In this lab you will work with a partner to experiment with some of the features of NFS. One of you will function as an NFS server, and the other will function as an NFS client. You should work *together* throughout the lab to ensure that you feel comfortable with both the client and server functionalities of NFS. At this point, decide between yourselves who will be the server and who will be the client.

Host name of server: _____

Host name of client: _____

Preliminary Steps

1. **(client)**
   Install the lab files needed on your client:

   ```
   # cd /labs
   # tar -xvf nfs.client.tar
   ```

   You should now have two new user accounts defined in your **/etc/passwd** file: "mickie" and "minnie." The passwords for the new accounts are "mickie" and "minnie" respectively. Note that neither user has a home directory on your machine. You will mount their home directories from your partner's NFS server.

2. **(server)**
   Install the lab files needed on your server:

   ```
   # cd /labs
   # tar -xvf nfs.server.tar
   ```

   This tarball creates several new files and directories, and two new user accounts in your **/etc/passwd** file for users "mickie" and "minnie." The passwords for the new accounts are "mickie" and "minnie" respectively. The tarball also creates home directories for mickie and minnie.

Part 1: Basic NFS Configuration

1. **(client and server)**
   In order for NFS to function properly, the InternetSrvcs and Networking products must be installed on your machine. Check to ensure that both of these products have been installed on your machine. Also ensure that the NFS subsystem is configured in the kernel.

   ```
   # swlist -l product 'Networking' 'InternetSrvcs' 'NFS'
   # grep nfs /stand/system
   ```


© 2003 Hewlett-Packard Development Company, L.P.

2. **(client and server)**
   Is your machine configured as an NFS server, client, or both?

   What configuration file should you check to find out?

   Make sure the appropriate functionality is configured.

   Answer

   Your machines should be configured with both NFS server and NFS client functionality.

   Check the NFS_SERVER and NFS_CLIENT variables in **`/etc/rc.config.d/nfsconf`**.

3. **(client)**
   What daemons should you see on an NFS client?

   Use **`ps -e`** on the client to ensure that the necessary daemons are actually running.

   Answer

   Clients should have the following daemons:

   `portmap`/**`rpcbind`** (optional)

   `biod` (optional)

   `rpc.statd`

   `rpc.lockd`

4. **(server)**
   What daemons should you see on an NFS server?

   Use **`ps -e`** to ensure that the server has the necessary daemons running.

   Answer

   Servers should have the following RPCs registered:

   `portmap`/**`rpcbind`**

   `rpc.mountd`

   `nfsd`

   `rpc.statd`

   `rpc.lockd`

   `rpc.pcnfsd` (optional)

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Exporting and Mounting NFS File Systems

1. **(server)**
   There are several files on your server machine that your clients need to access. Export the following with the export options set as noted. Make the file systems available to clients immediately, but also ensure that they will be available after the next system boot by adding them to **/etc/exports**.

   ```
   /home         rw for your partner's machine, no access for other hosts
   /opt/phone    rw for your partner's machine, readonly for all others
   /opt/fun      readonly for everyone on the LAN
   ```

   Answer

   server

   ```
   # vi /etc/exports
   /home       -access=client
   /opt/phone  -rw=client
   /opt/fun    -ro
   ```

   server

   ```
   # exportfs -a
   ```

2. **(server)**
   What command can you use to see what file systems you have made available? Can you tell which export options you used?

   What command can you use to see what file systems other servers have made available? Choose another machine in the classroom and see what it has exported. Can you tell which export options were used?

   Answer

   ```
   server# exportfs
   ```

   The **exportfs** command shows what is exported, and which export options were used.

   ```
   server# showmount -e otherservername
   ```

   This command shows who can mount the file system, but does *not* indicate what export options were used.

3. **(client)**
   Create mount points for the file systems your neighbor exported, and mount them:

   ```
   /home/mickie
   /home/minnie
   /opt/fun
   /opt/phone
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Answer

```
client# mkdir /home/mickie /home/minnie /opt/fun /opt/phone

client# mount server:/home/mickie /home/mickie

client# mount server:/home/minnie /home/minnie

client# mount server:/opt/fun /opt/fun

client# mount server:/opt/phone /opt/phone
```

4. **(client)**
What file needs to be modified to ensure that these NFS file systems are automatically mounted after every system boot? Make it so. (For now, use the "defaults" mount option.)

   Syntax errors in the **/etc/fstab** file may cause the next system boot to fail. Do a **mount -a** to ensure that you did not make any mistakes in **fstab** file.

   Finally, use **mount -v** to ensure that all the NFS file systems actually mounted properly.

   Answer

```
client# vi /etc/fstab
server:/home/mickie /home/mickie nfs defaults 0 0
server:/home/minnie /home/minnie nfs defaults 0 0
server:/opt/fun     /opt/fun     nfs defaults 0 0
server:/opt/phone   /opt/phone   nfs defaults 0 0

client# mount -a

client# mount -v
```

5. **(server)**
What command lists the remote machines that have your exported file systems mounted?

   Answer

```
server# showmount -a
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Using NFS File Systems

1. **(client and server)**
   Some shops use NFS to export file systems containing application executables. This offers a number of benefits. You only need to allocate disk space for the application on the NFS server, not on each and every client. It also simplifies upgrades, since the application is stored in just one place.

   From your client, try executing some of the programs mounted from the NFS server to verify that this is true:

   ```
   client# /opt/fun/melt
   client# /opt/fun/xroach -speed 1
   ```

   Another benefit of NFS is that files created in an NFS file system instantly become available to multiple client machines. Do the following experiment to verify that this is true:

   ```
   client# ls /home/mickie
   server# touch /home/mickie/data
   client# ls /home/mickie
   ```

   Does the client see the new file that was created on the server?

   Answer

   Yes, the client should see the new file that was created on the server.

2. **(client and server)**
   Though access to files shared via NFS should be more or less transparent to your users, file access restrictions can mean that a user is able to access a file on some machines but not others. Try the following commands while logged on as root:

   ```
   client# cp /opt/fun/melt /opt/fun/drip
   server# cp /opt/fun/melt /opt/fun/drip
   ```

   Why did this command succeed when executed on the server, but not when executed on the client? (hint: look at **`/etc/exports`**)

   Answer

   The file system was exported with write permission, but without root permission. Thus, on the client, user root is treated as "nobody" inside **`/opt/fun`**.

3. **(client)**
   Let's try a variation on the experiment you did back in Q#1 of this part of the lab.

   ```
   client# touch /home/mickie/memo
   ```

   Why did this fail?

   Was the file system exported with "ro" permission?

© 2003 Hewlett-Packard Development Company, L.P.

Was the file system mounted with "ro" permission?

As root, shouldn't you be able to create **/home/mickie/memo**?

Do whatever is necessary to successfully execute the touch command on the client. (You should not have to type anything on the server. Hint: Which user on the client has write permission on Mickie's home directory?)

Answer

The file system was neither exported nor mounted with read only permissions. However, root on the client is treated as nobody. User nobody only has r-x permissions on mickie's home directory.

```
client# su - mickie
client# touch /home/mickie/memo
exit
```

4. **(client and server)**
We saw in the previous question that root on an NFS client does not (by default) have the same file access as root on the NFS server. If a single administrator manages several systems, however, it may be useful to allow root on NFS clients to have true root access to exported file systems.

   What would you have to do on the NFS server side to allow root on the client to have the same full root access to the **/home** file system? Make it so.

   Did this seem to work? While logged in as root on the client, try touching a file in mickie's home directory. Did you have to do anything on the client side to recognize the change in the server's **exports** file?

   Answer

```
server# vi /etc/exports
           /home -root=client
server# exportfs –a
client# touch /home/mickie/junk
```

   This should work, even without remounting the file system

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Unmounting NFS File Systems

1. **(client)**
   Occasionally, it becomes necessary to unmount file systems to perform some administrative tasks. Let's start with the easiest case: on the client machine, unmount **/home/mickie**.

   Use **mount -v** to see which file systems remain in the client's mount table. Also do an **ls** of **/home/mickie**, and note that the memo and data files that were under **/home/mickie** no longer appear since the file system has been unmounted.

   Answer

   ```
   client# umount /home/mickie

   client# mount -v
   client# ls /home/mickie
   ```

2. **(client)**
   Let us try a more complicated scenario. Can the client unmount an NFS file system if one of the client's users is accessing that file system?

   On the client machine, open two windows. In one of the windows, **cd** to the **/home/minnie** directory. In the other window, issue the **umount** command to unmount the minnie file system. Did this work?

   The **fuser** command can tell you who is currently using a file system. Try the following to see who is currently using /**home/minnie**.

   ```
   client# fuser -cu /home/minnie
   ```

   Try a **fuser -cuk** on **/home/minnie**, and see what happens. Then try unmounting the file system again.

   Answer

   This should fail. You cannot unmount a file system on the client while a process on the client is using the file system.

   This command kills the window that was using **/home/minnie**. After killing the process, its no problem to **unmount** the file system.

   ```
   client# fuser -cuk /home/minnie
   client# umount /home/minnie
   ```

3. **(server)**

   In Part 2, Question 5, we saw a command that the server administrator could use to determine which of the exported file systems were actually mounted on client hosts. Now try executing that command again. Was the NFS server notified when the client

© 2003 Hewlett-Packard Development Company, L.P.

unmounted mickie and minnie?

Answer

```
server# showmount -a
```

The output suggests that the server was notified.

4. **(server and client)**

We saw that the administrator can force users out of a mounted file system with the **fuser** command. If **fuser** is executed on the NFS server, does it kill processes on the NFS clients, or just on the server itself? Try it.

```
client# cd /opt/fun
server# fuser -cuk /opt
```

Unfortunately, there is no mechanism in NFS to kill client processes from the server.

Answer

You should see that the **fuser** command, when executed on the server, only kills processes on the server. The clients should be unaffected.

5. **(server and client)**

We just discovered in the previous question that the NFS server has no way of killing processes on client hosts. Local file systems cannot be unmounted until all processes using them die. Does this mean that an NFS server administrator is unable to unmount his/her exported file systems until the clients that have mounted those file systems voluntarily unmount? Let's find out.

```
server# fuser -cuk /opt  # kill any proc's on the svr using /opt
server# umount /opt      # unmount the local /opt file system
```

Did you successfully unmount the file system? Any errors? What happened to the client process that was using your exported **/opt**?

Try the following commands on the client and note the output.

```
client# pwd
client# ls
client# cd ..
client# cd /
client# umount /opt/fun
```

On the client, could you unmount **/opt/fun**, even after the server unmounted?

Answer

No errors. The client process does not appear to have been affected — yet.

© 2003 Hewlett-Packard Development Company, L.P.

pwd works.

ls generates: ". not found."

cd .. generates: "the specified directory is not valid."

cd / works.

umount /opt/fun works.

6. **(server and client)**
Summarizing what you saw in the previous question: If an NFS server unmounts an exported file system that a client has mounted,

   a. Can the client still access files in the effected file system?

   b. What happens to client processes accessing the affected file system?

   c. Can the client unmount the imported file system?

   Answer

   The client cannot access files in the effected file system.

   Processes using the file system are not affected initially.

   The client can unmount the file system.

7. **(server and client)**
Remount all the server and client file systems on both the server and client.

   Answer

```
server# mount -a
server# exportfs -a
client# mount -a
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 5: (Optional) When Things Go Wrong

1. During the remainder of the lab, you will be asked to shutdown your LAN card several times. Execute the following command to shutdown CDE before proceeding:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. **(server and client)**
   What happens if the NFS client loses LAN connectivity to the server? Do the following and note the output from the commands. First note the client's behavior when the server is up. (It should be normal.)

   ```
   client# cd /opt/fun
   client# ls
   ```

   Now take the server's lan card down and note what happens to the client:

   ```
   server# ifconfig lan0 down
   client# ls                    # This will hang indefinitely
   ```

   Move on to next step.

   What happens when the client regains connectivity to the NFS server?

   ```
   server# ifconfig lan0 up
   client# ls
   ```

   Answer

   This should work without any problems.

   The **`ls`** hangs indefinitely. Shortly, you should get an **`NFS server not responding`** error.

   The **`ls`** command executes.

3. **(server and client)**
   So, what can the client administrator do while the NFS server is down? Can the client administrator unmount the NFS file system? Try it.

   ```
   server# ifconfig lan0 down
   client# cd /
   client# umount /opt/fun       # Be patient.
   client# mount
   ```

   What happens if the client tries to remount that file system again while the server is still down? Try it.

   ```
   client# mount /opt/fun        # Be patient.
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Answer

The **umount** actually occurs immediately. However, the client attempts to notify the server that the file system has been unmounted. It may take several minutes for this to time out.

Eventually, the client should time out.

4. **(server and client)**
Hopefully you discovered that a client can always unmount an NFS file system, even if the NFS server is down. In fact, since NFS is a "stateless" system, the server can always unmount its local file systems, too, even if clients have them mounted. Of course doing so will cause problems for the clients.

   To summarize, when an NFS server goes down...

   Are any of the processes on the client killed?

   What happens when a process on the client tries to hit a file system on the downed server (assuming the default mount options are used)? Do they hang indefinitely or time out?

   What happens when a client tries to mount a file system from a downed server? (Again, assume that the default mount options are used.)

   Answer

   When the NFS server becomes unavailable, no client processes are killed. However, if a client process attempts to access the server, the process hangs indefinitely. The client can always unmount a file system, even if the NFS server is down.

5. **(server and client)**
Bring the server and client back to their original states:

```
server# mount -a
server# exportfs -a
client# mount -a
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Client Side Mounting Options

1. **(server and client)**
   intr and **nointr** mount options

   By default, HP-UX mounts NFS file systems "hard,intr." If the NFS server goes down with these default mount options, we saw client attempts to access the NFS files and directories hang indefinitely. Can the user abort a command if they get tired of waiting? Try it.

   ```
   server# ifconfig lan0 down
   client# ls /opt/fun          # can the user abort the ls with ^C?
   server# ifconfig lan0 up
   ```

   Alternately, you can mount an NFS file system **nointr**. How would the **nointr** mount option affect the experiment above? Try it.

   ```
   client# umount /opt/fun
   client# mount -o nointr server:/opt/fun /opt/fun
   server# ifconfig lan0 down
   client# ls /opt/fun                  # can the user abort the ls with ^C?
   ```

   When will the user get a prompt back?

   Answer

   With the default **intr** mount option, the user can ^C out of a process that hangs because of a downed NFS server.

   If the file system is mounted **nointr**, however, a process hung as the result of a downed NFS server hangs indefinitely. The user will get a prompt back only when it regains connectivity to the NFS server.

2. **(server and client)**
   Soft versus hard mounts
   The client can also override the **hard** option with **mount -o soft**. If a client has mounted an NFS file system "soft" and the NFS server goes down, what happens to client requests to the server? Try it.

   ```
   server# ifconfig lan0 up
   client# umount /opt/fun
   client# mount -o soft server:/opt/fun /opt/fun
   server# ifconfig lan0 down
   client# ls /opt/fun         # be patient.
   ```

   Answer

   Eventually, **ls** times out with a message saying: "NFS access failed." In contrast to this behavior, the "hard" option would have hung indefinitely.

© 2003 Hewlett-Packard Development Company, L.P.

Part 7: (Optional) Troubleshooting a Non-responsive NFS Server

1. **(client and server)**
   You have seen the effect that a downed NFS server has on NFS clients. What can the administrator on the client side do to determine what might be wrong on the server side? Do an experiment to find out. Start by doing some experiments while both your server and client are functioning properly. Bring the LAN cards on both machines to an "UP" state:

   ```
   server# ifconfig lan0 up
   client# ifconfig lan0 up
   ```

   Now test connectivity from the client to your NFS server:

   ```
   client# ping server
   client# rpcinfo -p server
   ```

2. **(server and client)**
   Now, shut down NFS on your NFS server.

   ```
   server# /sbin/init.d/nfs.server stop
   ```

   From the client, try mounting **/opt/fun**.

   ```
   client# mount /opt/fun # After you see the error, hit ^C
   ```

   Answer

   mount notes that the server's "mount" RPC is not registered.

3. **(client)**
   From the client, try your connectivity test commands again:

   ```
   client# ping server
   client# rpcinfo -p server
   ```

   Can you still **ping** the server?

   Which RPC programs are no longer available on the server?

   Will clients be able to mount NFS file systems now? Why?

   Will clients be able to access already mounted NFS file systems? Why?

   Answer

   You can still **ping** the server, but the **mount** and **nfs rpcs** are no longer registered with the server's portmapper. The client cannot mount the file system since **mountd** daemon is not running on the server anymore. Nor can the client access already mounted NFS file systems since the **nfs** daemon is not running.

© 2003 Hewlett-Packard Development Company, L.P.

Part 8: Cleanup

1. Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

10–15. LAB: Configuring AutoFS

Preliminary

This lab assumes that the classroom has been configured with the 128.1.*.* IP addresses configured earlier in the course. The instructor station must be assigned IP address 128.1.0.1. Execute the following preliminary setup steps on both the student and instructor workstations in preparation for the lab:

```
# /labs/autofs.lab.setup.sh
```

These scripts added several entries to the **/etc/passwd** and **/etc/hosts** files on both the instructor and student workstations. When executed on the instructor station, the script also configures several additional IP addresses via IP multiplexing, and creates and exports several directories.

Part 1: Enabling and Starting AutoFS

Before you can configure the AutoFS maps, you must verify that NFS is installed, and the AutoFS daemons are running. That is the goal of this first portion of the lab!

1. Verify that the NFS product is installed on your system, and that the NFS client functionality is configured in **/etc/rc.config.d/nfsconf**.
   Answer

   ```
   # swlist -l product NFS
   # more /etc/rc.config.d/nfsconf
   ```

   Make sure that the NFS_CLIENT variable is set to "1"!

2. AutoFS was not included in the NFS product that was initially shipped with 10.20 and 11.00. Verify that AutoFS is included in the version of the NFS product installed on your system by checking for the existence of the **/usr/lib/netsvc/fs/autofs** directory.

   Answer

   ```
   # ll /usr/lib/netsvc/fs/autofs
   ```

3. HP-UX 10.20 and 11.x support both AutoFS and the older Automounter. Is either of these services configured on your machine? Which one, if any?

   Answer

   ```
   # more /etc/rc.config.d/nfsconf
   ```

   If **AUTOMOUNT=0** and **AUTOFS=0**, then neither service is configured.
   If **AUTOMOUNT=1** and **AUTOFS=0**, then automounter is configured.
   If **AUTOMOUNT=1** and **AUTOFS=1**, then AutoFS is configured.

   Automounter is configured by default in both HPUX 10.20 and 11.00.

© 2003 Hewlett-Packard Development Company, L.P.

4. Enable AutoFS in **/etc/rc.config.d/nfsconf**, but do not try to start the daemon yet.

   Answer

   ```
   # vi /etc/rc.config.d/nfsconf
     AUTOMOUNT=1
     AUTOFS=1
   ```

   There is no need to change the defaults for any of the other AutoFS and Automount related variables in the file at this point.

5. Automount and AutoFS should never run concurrently on a system. Technically, you should be able to switch from one service to the other by tweaking the control variables in **/etc/rc.config.d/nfsconf**. Realistically speaking, however, it is often difficult to shut down automounter without rebooting since the daemon will not die until all of the automounted file systems are unmounted. The cleanest solution is to reboot. Make it so!

   ```
   # shutdown -ry 0
   ```

6. When your system comes back up again, verify that the AutoFS daemons are running.

   Answer

   ```
   # ps -ef | grep auto
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring the AutoFS –hosts Map

The **–hosts** map provides a convenient mechanism for automatically mounting NFS file systems from any NFS server without modifying **/etc/fstab** or issuing the **mount** command. This portion of the lab walks you through the steps required to configure the **–hosts** map.

1. The **–hosts** entry is included in **/etc/auto_master** by default in HPUX. Verify that the map has already been configured in your system's **/etc/auto_master** file.

   Answer

   Your **/etc/auto_master** file should look like this:

   ```
   # cat /etc/auto_master
     /net –hosts –nosuid,soft
   ```

2. Does the mount table reflect the fact that AutoFS is managing the **/net** mount point?

   Answer

   ```
   # mount –v
   ```

   Yes! You should see an entry in your mount table showing that **–hosts** is mounted on **/net**. The file system type field in the mount table should indicate that this is an **autofs** file system.

3. Test your **–hosts** map! What happens when you access **/net/corp**? Try it!

   ```
   # ls /net/corp
   ```

   Answer

   Several NFS file systems should have been mounted under **/corp** on your behalf, and should appear in the **ls** output.

4. What changed in the mount table?

   Answer

   ```
   # mount –v
   ```

   The **–hosts** entry in the mount table remains. Also, you should see one entry in the mount table for each of the NFS file systems mounted under **/net/corp/*** .

5. Will AutoFS recognize a host referenced by IP address rather than name? Try it!

   ```
   # ls /net/128.1.0.1
   # mount –v
   ```

   It works! You may reference hosts under **/net** by either host name or IP address.

© 2003 Hewlett-Packard Development Company, L.P.

6. What happens if you attempt to access a non-existent host? Try it!

```
# ls /net/10.1.1.1
```

Answer

The resulting AutoFS mount request fails, and AutoFS returns a "not found" message.

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring the AutoFS Direct Map

This part of the lab exercise gives you an opportunity to supplement your –hosts special map with a direct map file, too.

1. Add a direct map entry to **/etc/auto_master**. Name your direct map **/etc/auto.direct**.

   Answer

   ```
   # vi /etc/auto_master
     /-   /etc/auto.direct
   ```

2. Configure your direct map to automatically mount the **/usr/contrib/games** directory from the **corp** NFS server. Use the read-only mount option.

   Answer

   ```
   # vi /etc/auto.direct
     /usr/contrib/games –ro corp:/usr/contrib/games
   ```

3. What must be done to make this change take effect? Make it so!

   Answer

   ```
   # automount
   ```

4. What appears in the mount table to indicate that AutoFS has recognized the new direct map?

   Answer

   ```
   # mount –v
   ```

   There should be an entry in the mount table indicating that **/etc/auto.direct** is mounted on **/usr/contrib/games**.

5. Does the **games** mount point appear when you list the contents of **/usr/contrib**? Does listing the **/usr/contrib** directory cause AutoFS to mount the **games** file system from the NFS server?

   ```
   # ls /usr/contrib
   # mount –v
   ```

   Answer

   The **games** mount point directory does appear in the **ls** output. However, the file system does not actually mount until the contents of **/usr/contrib/games** are first accessed.

© 2003 Hewlett-Packard Development Company, L.P.

6. **cd** to **/usr/contrib/games**, and list the contents. There should be an executable under **games** called **/usr/contrib/games/oneko/bin/X11/oneko**. Run the **oneko** executable, then check the mount table to see what changed.

```
# cd /usr/contrib/games
# ls
# /usr/contrib/games/oneko/bin/X11/oneko &
# mount –v
```

Answer

Any attempt to access the contents of an AutoFS managed mount point should cause the associated NFS file system to mount. Any one of the three actions in this question – **cd**, **ls**, or running the executable – would have been sufficient to cause the file system to mount.

Viewing the mount table should verify this. You should see **/usr/contrib/games** mounted from the NFS server.

7. Add another entry to your direct map to mount the **/data/contacts** directory from the corp NFS server. Users will need both read and write access to this file system. Do not execute the **automount** command yet.

Answer

```
# vi /etc/auto.direct
  /usr/contrib/games –ro corp:/usr/contrib/games
  /data/contacts     -rw corp:/data/contacts
```

8. What happens at this point if you attempt to do an **ls** of **/data/contacts**?

Answer

```
# ls /data/contacts
```

This should generate a "not found" error message. The **automount** command must be executed to notify AutoFS any time the direct map changes.

9. Do whatever is necessary to make the **/data/contacts** directory available on the client. Verify that your fix works.

Answer

```
# automount
# mount -v
# ls /data/contacts
# mount -v
```

This time, the **ls** command should succeed!

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Configuring an AutoFS Indirect Map

Your organization has three departments, with home directories on three different NFS servers. Members of the finance department have their home directories on a server called "finance," members of the business department have their home directories on a server called "business", and members of sales have their home directories on a server called "sales". Your goal in this portion of the lab exercise is to configure an indirect map that will mount and unmount these home directories on an as-needed basis.

1. The indirect map used in this portion of the lab will be mounted under **/home**. This will not work if the logical volume containing your current users home directories is also mounted on **/home**. For the remainder of this lab, unmount the logical volume containing your users' home directories.

   ```
   # umount /home
   ```

2. Add an indirect map entry for **/home** to **/etc/auto_master**. This map entry should reference the **/etc/auto.home** map file.

   Answer

   ```
   # vi /etc/auto_master
     /home /etc/auto.home
   ```

3. What must be done anytime the master map changes? Make it so!

   Answer

   You must update the mount table anytime the master map changes:

   ```
   # automount
   # mount -v
   ```

4. Now create the **/etc/auto.home** map file. The map file should configured such that:

   - **/home/finance** is mounted from **finance:/home/finance**
   - **/home/business** is mounted from **business:/home/business**
   - **/home/sales** is mounted from **sales:/home/sales**

   Is it necessary to re-issue the automount command after creating/changing the indirect map file?

   Answer

   ```
   # vi /etc/auto.home
     finance   finance:/home/finance
     business  business:/home/business
     sales     sales:/home/sales
   ```

   It is *not* necessary to execute **automount** after modifying an indirect map. This is one key advantage that the indirect map has over a direct map!

© 2003 Hewlett-Packard Development Company, L.P.

5. Check the mount table. How many mount table entries were created because of the new indirect map? How many entries would have been created in the mount table if this had been configured as a direct map?

   Answer

   ```
   # /usr/sbin/mount –v
   ```

   There should be just one new entry in the mount table indicating that **/etc/auto.home** is mounted on **/home**. If this had been configured via a direct map, there would have been three new entries in the mount table.

6. Do an **ls** of **/home**. Can you explain the result? Did AutoFS mount any file systems?

   Answer

   ```
   # ls /home
   # mount -v
   ```

   The **ls** command does not list anything! This is expected. The home directories will not be mounted until they are actually accessed.

7. Now access a specific user's home directory and see what happens to the mount table:

   ```
   # ls /home/finance/user1
   # mount –v
   ```

   Answer

   AutoFS intercepts the **/home/finance** access attempt, and automatically mounts the needed file system from the finance server. This is reflected in the mount table.

8. Will this configuration automatically mount a user's home directory at login time? Try it! Try logging in as user "user3." Then check the mount table to verify that the user's home directory was in fact mounted from the proper location.

   ```
   # su – user3
   $ pwd
   $ ls -a
   $ exit
   # mount -v
   ```

   Answer

   The user login should succeed. The login process attempts to cd to the home directory specified by the user's entry in the **/etc/passwd** file. Assuming **/etc/passwd** and AutoFS are configured properly, users will never know that their home directories are mounted by AutoFS.

© 2003 Hewlett-Packard Development Company, L.P.

9. Can you shorten the **/etc/auto.home** file to a single line? How? Make it so! Then test your solution:

```
# vi /etc/auto.home
# ls /home/sales/user5
# mount –s.
```

Answer

```
# vi /etc/auto.home
  * &:/home/&
# ls /home/sales/user5
# mount -v
```

AutoFS key substitution provides the solution to this problem. The **/etc/auto.home** file suggested below will automatically NFS mount any user's home directory if each NFS server's home directories are named according to the following convention: **/home/*servername*/username**. The **ls** command should succeed.

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Cleanup

Before moving on to the next chapter, run the **netfiles.sh** cleanup script:

```
# /sbin/init.d/nfs.client stop
# mount -a
# /labs/netfiles.sh –r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

11–14. LAB: Configuring NIS

Directions

In this lab exercise, you will work with a team of two to four classmates to configure and test NIS servers and clients in your own NIS domain. Working with the teammates assigned by your instructor, decide on a name for your NIS domain.

Domain Name: _____

Within your domain, you should configure one master server, a slave server, and one or more clients. Decide among yourselves which machine will be your master server, which will be the slave, and which will be the client(s):

Master server: _____

Slave server: _____

Client(s): _____

Note that the examples referenced in the instructions that follow refer to a domain called "california" containing three hosts. Within this sample domain, "sanfran" is the master server, "oakland" is the slave server, and "la" is a client.

Part 1: Configuring an NIS Master Server

The following steps should only be performed on the NIS master server. Do not start configuring the slave or clients until the master configuration is complete.

1. Ensure that your ASCII source files (**/etc/passwd**, **/etc/group**, etc.) are up-to-date. Although the ASCII files may be changed after configuring NIS, it is much easier to make changes now. For the sake of this lab exercise, you may assume that your ASCII source files are already up-to-date.

2. The script used to configure the NIS master server must know ahead of time the name of the domain. Do this by setting your server's NIS domain name with the **domainname** command:

```
# domainname california   # set your domain name
# domainname              # check your domain name
```

3. Next, run the **ypinit -m** command to build all the maps for your domain. When asked if you wish to "quit on non-fatal errors," Answer "**n**." **ypinit** prompts for a list of slave servers for the domain, then builds all the necessary maps.

```
# ypinit -m
```


© 2003 Hewlett-Packard Development Company, L.P.

4. Next, modify the NIS startup configuration file, **/etc/rc.config.d/namesvrs**. Enable your machine as an NIS_MASTER_SERVER and define your NIS_DOMAIN. To ensure consistency across the domain, the master should also be configured as a client. Enable NIS_CLIENT functionality as well.

   Answer

   ```
   # vi /etc/rc.config.d/namesvrs

   NIS_MASTER_SERVER=1

   NIS_SLAVE_SERVER=0

   NIS_CLIENT=1

   NIS_DOMAIN=california
   ```

5. Reboot to start NIS on the master.

   Answer

   ```
   # cd /
   # shutdown -ry 0
   ```

6. When your machine comes back up again, check to see which processes are running. What NIS-related processes would you expect to see on an NIS master server?

   Answer

   Among others, you should see **portmapper/rpcbind**, **ypserv**, **rpc.yppasswd**, and **ypbind**. A complete list of NIS-related daemons was provided earlier in the chapter.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring an NIS Slave Server

Every NIS domain should have at least one NIS slave server to provide service to the clients if the master becomes unavailable. In subnetted networks, each subnet usually has a separate NIS slave server.

Do not begin this portion of the lab until the master server is fully configured.

1. Start by setting your domain name as you did on the master.

   Answer

   ```
   # domainname california
   ```

2. Run the **ypinit -s *masterserver*** command, where ***masterserver*** is the host name of your master server. This downloads the NIS maps from the master. When asked if you wish to quit on non-fatal errors, Answer "**n**."

   ```
   # ypinit -s sanfran
   ```

3. Watch the **ypinit** messages. What does the **ypinit** do to configure the slave server? (Note: disregard the ethers, bootparams, and netmasks errors generated by **ypinit**. These maps are not used in HP-UX, but the **ypinit** utility still attempts to download them.)

   Answer

   ypinit automatically downloads all the NIS maps from the master server.

4. **ypinit** should have copied the NIS maps from the master server, and stored them under the slave server's **/var/yp directory**. Do an **ls** of **/var/yp**, and find the subdirectory for your domain. What do you see in your domain's **/var/yp** subdirectory?

   Answer

   All NIS maps are stored in subdirectories under **/var/yp**. The california maps, for instance, would be found in **/var/yp/california**.

5. Next, modify the NIS startup configuration file, **/etc/rc.config.d/namesvrs**. Enable your machine as an NIS_SLAVE_SERVER and NIS_CLIENT and define your DOMAINNAME.

   Answer

   ```
   # vi /etc/rc.config.d/namesvrs

   NIS_MASTER_SERVER=0

   NIS_SLAVE_SERVER=1

   NIS_CLIENT=1

   NIS_DOMAIN=california
   ```


© 2003 Hewlett-Packard Development Company, L.P.

6. Remove all of your users' entries from your local password file, since NIS will now be providing central administration of your user account information. However, be sure to leave all accounts with userids below 100 in **/etc/passwd**. Why might it be important to leave these userids (especially root.) in place?

   Answer

   ```
   # vipw              # remove all user account definition lines
   ```

   If there are problems with NIS, you should ensure that at least the critical system accounts are still available so root can log on and fix the problem.

7. Reboot.

   Answer

   ```
   # cd /

   # shutdown -ry 0
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring NIS Clients

Do not continue on to this step until at least one of your NIS servers has finished booting. Now configure the remaining hosts in your team as NIS clients.

1. Enable NIS client functionality and define your domain name in the **/etc/rc.config.d/namesvrs config** file.

   Answer

   ```
   # vi /etc/rc.config.d/namesvrs

   NIS_CLIENT=1

   NIS_DOMAIN=california
   ```

2. As you did with your slave server, remove all user entries from **/etc/passwd**.

   Answer

   ```
   # vipw
   # remove all user entries, but leave userid's 0 -100
   ```

3. Reboot.

   Answer

   ```
   # cd /

   # shutdown -ry 0
   ```


© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Using NIS Maps

After the system finishes booting, try a few tests to see if your NIS configuration was successful. Since all of your machines in the domain are clients, even the master and slave can try these exercises.

1. The **ypwhich command** tells you to which server you are bound. Which server are you currently bound to?

   Answer

   ```
   # ypwhich
   ```

2. The **ypcat** command displays the contents of NIS maps. Adding the **-k** option also shows the key value associated with each entry in the map files. View the contents your **hosts** map by typing:

   ```
   client# ypcat -k hosts.byname
   client# ypcat -k hosts.byaddr
   client# ypcat hosts
   ```

   "**hosts**" is just an abbreviation for **hosts.byaddr**. To list the other nicknames recognized by **ypcat**, try:

   ```
   client# ypcat -x
   ```

3. You can check the value associated with any key in an NIS map by using the **ypmatch** command:

   ```
   client# ypmatch user1 passwd.byname
   client# ypmatch 0  passwd.byuid
   ```

4. Do the standard UNIX utilities use the NIS? To find out, try logging in as user1. Note that user1 no longer exists in the slave or clients' local password files. Why does this login succeed?

   Answer

   ```
   client# login user1      # login as user1
   client# exit             # log back out again
   ```

   The system calls used to look up usernames and passwords are smart enough to reference the NIS maps instead of the local password file.

5. Try another system utility. Use **nslookup** to determine which IP address is associated with your neighbor's host name. Does **nslookup** appear to use NIS? How can you tell?

   Answer

   ```
   client# nslookup oakland
   ```

   nslookup notes in its output: **"Trying NIS."**

   Even if the **/etc/hosts** file did not exist, your client could resolve host names using the NIS hosts map.

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Updating NIS Maps

1. Start with an easy NIS update. Log in as user1 on the client and type **passwd** to change user1's password.

   Answer

   ```
   client# login user1        # login as user1

   client# passwd             # change user1's password

   client# exit
   ```

2. Is the password change reflected in the password map on the master, the slave, or both? Use **yppoll** command to check the order number on the master and slave servers.

   ```
   yppoll -h slave passwd.byname
   yppoll -h master passwd.byname
   yppoll -h slave passwd.byuid
   yppoll -h master passwd.byuid
   ```

3. Are the order numbers the same?

   Answer

   The order numbers should be the same, which indicates that both servers' maps were updated.

4. Try another change on the client. Create a user account in the **/etc/passwd** file on the client, then **ypcat** the passwd map again. Does **ypcat** show the new account? Explain.

   ```
   client# useradd donald
   client# ypcat passwd
   ```

   Answer

   ypcat does not reflect the changes. NIS consults the NIS maps (which haven't changed yet) rather than the local **passwd** file.

5. What happens if you make your changes to **/etc/passwd** on the master server instead of the client? Try it. Add user donald to the master server's passwd file. Then **ypcat** the passwd map and explain the result.

   ```
   master# vi /etc/passwd
   master# ypcat passwd
   ```

   Answer

   Even changing the ASCII source files on the master will not yield an immediate change in the ypcat output. Remember, ASCII source files are distinct from NIS maps. The master's NIS maps must be rebuilt and pushed out to the slaves anytime the ASCII source files change.

© 2003 Hewlett-Packard Development Company, L.P.

6. On the master, do whatever is necessary to rebuild the passwd map and propagate the updates to the slave server. Use **ypcat** to ensure this worked properly.

   Answer

   ```
   master# /var/yp/ypmake passwd
   master# ypcat passwd
   ```

   ypmake rebuilt both password maps and automatically pushed them out to the slave.

7. What happens if an NIS slave is down when the master attempts to push an update? Try it and find out.

   - Shutdown the LAN card on the slave
   - Add user **pluto** to the master's **/etc/passwd** file
   - **ypmake** the passwd map on the master (Be patient.)

   Did **ypmake** warn you that the slave was down?

   Answer

   ```
   slave# ifconfig lan0 down
   master# useradd pluto
   master# ypmake passwd
   ```

   ypmake should display a "Timeout talking to slave" warning. However, the final message from **ypmake** says: "no errors encountered." Make a habit of reading ALL the messages from **ypmake** so you do not miss timeout warnings.

8. Bring the slave's LAN card back up again, then do whatever is necessary on the slave to update the maps. Note: **ypxfr** does not recognize the NIS nicknames.

   Answer

   ```
   slave# ifconfig lan0 up
   slave# ypxfr passwd.byuid
   slave# ypxfr passwd.byname
   ```

9. Is any harm done if you **ypxfr** a map that is already up-to-date? Try it. From the slave, try another **ypxfr** on **passwd**. What happens? Why might this behavior be advantageous?

   Answer

   ```
   slave# ypxfr passwd.byname
   slave# ypxfr passwd.byuid
   ```

   ypxfr only downloads new copies of the maps if there have been changes. Since the maps on the master have not changed since the last **ypxfr**, there was no need to download the maps again. The slave's maps remain unchanged.

© 2003 Hewlett-Packard Development Company, L.P.

Part 6: Securing Clients and Slave Servers with Password/Group Escape Entries

Currently, anyone listed in the NIS passwd map can log onto your NIS client. Your goal in this exercise is to modify your client configuration so only user1-user3 are allowed to log in (as well as root, of course).

1. Start out by adding the escape entries to the client's **/etc/passwd** file that would allow user1-user3, but no other NIS map users, to successfully log in.

   Answer

   ```
   client# vipw          # add the following lines to the end of the file

   +user1
   +user2
   +user3
   ```

2. Did your escape entry have the desired effect? Can your client su to user1's account? Can your client su to user6's account? Why can user6 still log in?

   Answer

   Both users appear to be able to log in despite the escape entry. By default, HPUX 11.x does not recognize escape entries. In order to force the system to recognize the escape entries, you must modify `/etc/nsswitch.conf`.

3. Create a new **/etc/nsswitch.conf** file for yourself with the entries required to recognize escape characters in **/etc/passwd** and **/etc/group**.

   Answer

   ```
   client# vi /etc/nsswitch.conf
   passwd: compat
   group: compat
   ```

4. Try logging in with the user1 and user6 usernames again. What happens now?

   Answer

   ```
   client# su - user1    # succeeds.
   client# su - user6    # fails.
   ```

   This is the desired behavior.

5. Change your client's password file so all users in the NIS maps **except** user1 -- user3 are allowed to log in. Try logging in.

   Answer

   ```
   client# vipw
    - user1
    - user2
    - user3
   +
   client # su - user1            # fails.
   client# su - user6 # succeeds.
   ```


© 2003 Hewlett-Packard Development Company, L.P.

This seemed to work.

Part 7: (Optional) Securing the NIS Master Server

The escape entries you used in the previous part of the exercise provide a convenient mechanism for restricting access to NIS clients and slaves. However, some special NIS configuration changes are required if you wish to restrict access to the master server.

1. Why can't you restrict access to the master server by simply deleting all the user lines from **/etc/passwd**, so only the root and basic system userids remain?

   Answer

   The **/etc/passwd** file on the master is used to build the passwd map. Deleting all the user lines would leave the passwd maps empty after the next **ypmake**.

2. Follow the steps suggested in the notes to restrict access to the master server so only root can log in.

   Answer

   ```
   master# cp /etc/passwd /etc/passwd.nis
   master# vipw
   ```

   (Remove all user entries)

   ```
   master # vi /etc/nsswitch.conf
   passwd: compat
   group: compat

   master# vi /etc/rc.config.d/namesvrs
   ```

   (Find the **YPPASSWDD_OPTIONS** line)

   (Change all occurrences of **/etc/passwd** to /**etc/passwd.nis**)

   ```
   master# vi /var/yp/ypmake
   ```

   (Change **PWFILE=${PWFILE:-$DIR/passwd}** to **PWFILE=${PWFILE:-$DIR/passwd.nis}**)

   ```
   master# /var/yp/ypmake passwd
   master# cd /
   master# shutdown -ry 0
   ```

3. Try logging into your master server as user3. This should fail.

© 2003 Hewlett-Packard Development Company, L.P.

Part 8: (Optional) When Things Go Wrong . . .

1. During the remainder of the lab, you will be asked to shutdown your LAN card several times. Execute the following command to shutdown CDE before proceeding:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. What happens if the NIS master server is unreachable for a period of time? Take down the LAN card on your master server.

   Answer

   ```
   master# ifconfig lan0 down
   ```

3. Can clients still access the maps? From the client, **ypcat** passwd and explain the result. (Be patient.)

   Answer

   ```
   client # ypcat passwd
   ```

   This should work. If the client was bound to the master, it may take a few minutes to timeout, but eventually **ypbind** should send out a broadcast to find a new server to which it can bind. The slave should be able to provide the requested map.

4. Can changes be made to the maps while the server is down? Log in as user4 on the client and try changing the password with **passwd**. What happens? (Be patient.)

   Answer

   ```
   client # login # log in as user4
   client # passwd
   client # exit
   ```

   The **passwd** command fails. No changes may be made to the maps until the master server returns.

5. Now take down the slave's LAN card, too.

   Answer

   ```
   slave# ifconfig lan0 down
   ```


© 2003 Hewlett-Packard Development Company, L.P.

6. Try a **ypcat** on **passwd**. What happens? (Be patient. Once you see a few error messages, press return to get back to a prompt.)

   Answer

   Eventually, you should get "NIS server not responding" messages, and the command times out with a failure message since no servers are available.

7. Bring the LAN cards on both servers back up again.

   Answer

```
master# ifconfig lan0 up
slave# ifconfig lan0 up
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 9: (Optional) Troubleshooting NIS

You have seen what happens when a client is no longer able to communicate with the NIS servers. What can you do to troubleshoot the problem?

1. What NIS-related process(es) must be running on the client? Do a **`ps -e`** to ensure that the necessary processes are actually running.

   Answer

   ```
   client # ps -e
   ```

   At a minimum, the client requires **`ypbind`**.

2. See if your client can still access the NIS maps. Try a **`ypcat passwd`** and see what happens (be patient).

   When an NIS server goes down, the client's first access may eventually time out and generate an error. However, **`ypbind`** immediately attempts to bind to another NIS server on the subnet. Try another **`ypcat passwd`** and see what happens. Did the **`ypcat`** succeed this time?

   Answer

3. There are a number of RPC daemons that must be running on an NIS server in order for clients to be able to access the NIS maps. How can the client see if these RPCs are registered and available?

   Answer

   ```
   client# rpcinfo -p sanfran     # test the master
   client# rpcinfo -p oakland     # test the slave
   ```

   The master server should be running **`rcbind`** (**`portmap`**), **`ypbind`**, **`ypserv`**, **`ypxfrd`**, **`yppasswd`**, and **`ypupdated`**.

   The slave should be running all of the above except **`rpc.yppasswd`** and **`rpc.ypupdated`**.

Part 10: Cleanup

Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

12–22. LAB: DNS

Introduction

In this exercise, you will configure a DNS master server, a slave server, and a DNS client. You will also have a chance to update the DNS data on your name servers, and explore some of the name server database files.

Your instructor will break the class into teams of 2 or 3 students each. Each team will be assigned a DNS sub-domain under **`hp.com`** from the table below. You will then work with your teammates to configure a master server, a slave server, and one or more DNS clients within your assigned domain. The instructor's station will serve as a root level name server so you can access other teams' domains as well.

Table 12-1.

Domain Name	Role	Host Name	IP Address
`hp.com`	master	corp	128.1.0.1
`ca.hp.com`	master	sanfran	128.1.1.1
	slave	oakland	128.1.1.2
	client	la	128.1.1.3
`il.hp.com`	master	chicago	128.1.2.1
	slave	peoria	128.1.2.2
	client	rockford	128.1.2.3
`ga.hp.com`	master	atlanta	128.1.3.1
	slave	athens	128.1.3.2
	client	macon	128.1.3.3
`ny.hp.com`	master	nyc	128.1.4.1
	slave	albany	128.1.4.2
	client	buffalo	128.1.4.3
`fr.hp.com`	master	paris	128.1.5.1
	slave	lyon	128.1.5.2
	client	grenoble	128.1.5.3
`uk.hp.com`	master	london	128.1.6.1
	slave	leeds	128.1.6.2
	client	ipswich	128.1.6.3
`de.hp.com`	master	bonn	128.1.7.1
	slave	berlin	128.1.7.2
	client	hamburg	128.1.7.3
`jp.hp.com`	master	tokyo	128.1.8.1
	slave	kyoto	128.1.8.2
	client	osaka	128.1.8.3

© 2003 Hewlett-Packard Development Company, L.P.

Part 1: Configure Your Master server

1. Delete all entries from the **/etc/hosts** file except the **localhost** entry and the hosts in your domain. Fully qualify all of the host names. The example below shows the changes that would be required on **sanfran**.

```
# vi /etc/hosts
127.0.0.1 localhost
128.1.1.1 sanfran.ca.hp.com sanfran
128.1.1.2 oakland.ca.hp.com oakland
128.1.1.3 la.ca.hp.com      la
```

2. Create a directory for the DNS database files and **cd** to it.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
# cd /etc/named.data
```

3. Create a **param** file for your domain.

```
# vi param
    -d ca.hp.com        # Use your domain name(s) here
    -n 128.1.1          # Use your subnet address(es) here
    -z 128.1.1.1        # Use your master server's IP here
    -b /etc/named.conf
```

4. Run **hosts_to_named**.

```
# hosts_to_named -f param
```

5. Copy the **db.cache** file from the instructor station.

```
# ftp 128.1.0.1
> get /etc/named.data/db.cache
> quit
```

6. Enable **NAMED** in the **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

7. Start the **named** daemon.

```
# /sbin/init.d/named start
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configure Your Slave server

1. Create a directory for the database and configuration files.

```
# mkdir /etc/named.data
# chmod 755 /etc/named.data
```

2. FTP copies of the **db.*** files from the master.

```
# ftp 128.1.1.1      # Use your master server's IP here
> mget /etc/named.data/db.*
> quit
```

3. FTP a copy of **conf.sec.save** from the master server, and move it into place on the slave server as **/etc/named.conf**.

```
# ftp 128.1.1.1       # Use your master server's IP here
> get /etc/named.data/conf.sec.save
> quit
# mv /etc/named.data/conf.sec.save /etc/named.conf
```

4. Enable **NAMED** in **/etc/rc.config.d/namesvrs**.

```
# vi /etc/rc.config.d/namesvrs
  NAMED=1
  NAMED_ARGS=""
```

5. Start the **named** daemon. A reboot is not necessary.

```
# /sbin/init.d/named start
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configure All Hosts (Including Your DNS Servers) in Your Domain as DNS Clients

1. Create/modify the resolver configuration file. Include your domain and the **hp.com** domain in your **search** list. Include both your master and your slave server in the **nameserver** list.

```
# vi /etc/resolv.conf
  search ca.hp.com hp.com # replace ca.hp.com with your domain name
  nameserver 128.1.1.1    # replace 128.1.1.1 with your master's IP
  nameserver 128.1.1.2    # replace 128.1.1.2 with your slave's IP
```

2. If your **/etc/nsswitch.conf** exists, delete it. You can experiment with the default behavior for now. You will have a chance to re-create the file later.

```
# rm /etc/nsswitch.conf
```

3. If you are the master server, skip this step! Slaves and clients do need to modify **/etc/hosts** at this point. Fully qualify and create aliases for your host in your local domain, and remove all other entries (except **localhost**).

```
# vi /etc/hosts
     127.0.0.1   localhost
     128.1.1.3   la.ca.hp.com  la
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Test DNS

All hosts in your domain (clients and servers) can try the following exercises.

1. Run **nslookup** and identify your master server as the server to use.

   - Can you resolve a host name in your own domain?
   - Can you resolve an IP address in your own domain?
   - Can you resolve a host name in another domain? (Try corp.hp.com.)
   - Can you resolve an IP address in another domain? (Try 128.1.0.1.)

   Answer

   The solutions below are for the "ca" domain.

```
# nslookup
> server 128.1.1.1    <-- Tell nslookup to use the master
> oakland.ca.hp.com.  <-- Resolve a host name in your host name
> corp.hp.com.        <-- Resolve corp.hp.com
> 128.1.1.2           <-- Resolve an IP in your domain
> 128.1.0.1           <-- Resolve corp's IP
> exit
```

   All of these tests should succeed.

2. Try the same tests that you did in the previous question, but use the slave name server this time. Does your slave server seem to work?

   Answer

```
# nslookup
> server 128.1.1.2    <-- Tell nslookup to use the slave
> sanfran.ca.hp.com.  <-- Resolve your host name
> corp.hp.com.        <-- Resolve another host in your domain
> 128.1.1.1           <-- Resolve your IP
> 128.1.0.1           <-- Resolve corp's IP
> exit
```

   All of these tests should succeed.

3. Which name server does **nslookup** use by default if you simply type **nslookup corp.hp.com** from the shell prompt? Try it. How can you permanently change the default name server?

   Answer

   The default name server is defined by the first **nameserver** entry in **/etc/resolv.conf**. Reversing the order of the **nameserver** entries in **/etc/resolv.conf** changes the default name server.

© 2003 Hewlett-Packard Development Company, L.P.

4. Try resolving a host name in your domain using the simple host name (that is, sanfran, rather than sanfran.ca.hp.com). Try resolving a host in another domain using the simple host name. Your first experiment should succeed, while the second should fail. Why?

   Answer

   ```
   on sanfran:
   # nslookup sanfran   (succeeds)
   # nslookup chicago   (fails)
   ```

   The second example fails since **`il.hp.com`** is not included in **`sanfran`**'s search list.

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Updating Your DNS Name Servers

1. Choose a new host/IP for your domain, and add it to your master server's DNS data files using vi and **hosts_to_named**. Do not run **sig_named**, yet. Note that you can add a new host name/IP to DNS even if that host has not been physically connected to the network yet.

   Answer

   The example that follows adds host **sacramento.ca.hp.com** to **sanfran**, the master server for **ca.hp.com**.

   ```
   # vi /etc/hosts
        127.0.0.1        localhost
        128.1.1.1        sanfran.ca.hp.com.
        128.1.1.2        oakland.ca.hp.com.
        128.1.1.3        la.ca.hp.com.
        128.1.1.4        sacramento.ca.hp.com.

   # cd /etc/named.data

   # hosts_to_named -f param
   ```

2. Which two **db.*** files would you expect to be affected by the newly added host and IP? Look at the SOA records for those two files. How can you tell that the files were updated?

   Answer

   Two files are affected by the addition of **sacramento**:

   ```
   /etc/named.data/db.ca
   /etc/named.data/db.128.1.1
   ```

   This is reflected by the serial number in the SOA records at the top of both files; the serial number has been incremented by one.

3. Now that the **db.*** files have been updated, can you **nslookup** the new host using the master server? Try it, and explain the results.

   Answer

   `# nslookup sacramento.ca.hp.com` (fails!)

   The **nslookup** command fails. **named** must be forced to reread its data files before it will resolve **sacramento**.

4. What do you need to do to ensure that your DNS clients can resolve the new host name? Make it so.

   Answer

© 2003 Hewlett-Packard Development Company, L.P.

Run **sig_named** on the master server to force the **named** daemon to reload its data files.

```
# sig_named restart
# nslookup
> server 128.1.1.1          ← Use your master server's IP here.
> sacramento.ca.hp.com      ← This should succeed!
> exit
```

5. By default, when will your slave name server recognize that a new host name and IP have been added to the domain? How can you force the slave to do an immediate update? Do it.

Answer

By default, the slave will only refresh its DNS data at the interval specified in the SOA records. Typically, the refresh interval is 3 hours. You can force an immediate refresh by restarting named on the slave server:

```
# sig_named restart
```

6. Verify that the slave server update was successful.

```
# nslookup
> server 128.1.1.2          ← Use your slave server's IP here.
> sacramento.ca.hp.com      ← This should succeed!
> exit
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Handling DNS Server Failures

1. This portion of the lab exercise asks you to take your LAN card down, and bring it back up again several times. Losing LAN connectivity will generally cause CDE to hang. Before attempting this portion of the lab exercise, shutdown CDE:

   ```
   # /sbin/init.d/dtlogin.rc stop
   ```

2. What happens if the master DNS server becomes unreachable? Can clients still resolve host names? Use the **ifconfig** command to take down the LAN card on your master server and see if your clients can still resolve **corp.hp.com**. (Be patient!)

   ```
   master# ifconfig lan0 down
   client# nsquery hosts corp.hp.com
   ```

   How is the client able to resolve host name corp if the master server is inaccessible?

   Answer

   The **/etc/resolv.conf** file lists two name servers. If one name server fails to respond, the resolver queries the slave name server, which should still be functioning at this point. The query to the second name server should succeed and respond back with corp's IP address.

3. What happens if the slave name server becomes unavailable, too? Try it. While the LAN card on the master server is still down, disable the LAN card on the slave server, too. Then attempt to resolve **corp.hp.com** again.

   ```
   master# ifconfig lan0 down
    slave# ifconfig lan0 down
   client# nsquery hosts corp.hp.com
   ```

   What happens this time?

   Answer

   nsquery hangs temporarily while waiting for responses from the two name servers, but should time out after two or three minutes. Unless you created an **/etc/nsswitch.conf** file, **nsquery** should fallback and try **/etc/hosts**. The client should still be able to resolve host names recorded in the **/etc/hosts** file – if you are patient!

4. Bring both servers' LAN cards back up again.

   ```
   master# ifconfig lan0 up
    slave# ifconfig lan0 up
   ```


© 2003 Hewlett-Packard Development Company, L.P.

5. Occasionally, you may have some development machines or other temporary hosts on your LAN. Rather than add these temporary host names to your DNS server databases, you may wish to simply record them in the local hosts file. Add a new host name/IP pair to your client's local hosts file. Then **nsquery** the new host name. What happens? Why?

   Answer

   Assuming that you have not created an **/etc/nsswitch.conf** file, nsquery will fail. The default switch configuration only consults **/etc/hosts** if DNS and NIS are UNAVAILable. In this case, since DNS yields "NOTFOUND," **nsquery** never consults **/etc/hosts**.

6. Change the client's hosts policy in **/etc/nsswitch.conf**. Configure your client's resolver to check the local **/etc/hosts** if DNS is unable to resolve a host name. Then **nsquery** the new host name you added to your client's **/etc/hosts** file in the previous step. What happens?

   Answer

```
client# vi /etc/nsswitch.conf
           hosts: files [ NOTFOUND=continue ] dns
client# nsquery hosts sacramento.ca.hp.com
```

   The nsquery should succeed this time, and indicate that the result came from the local **/etc/hosts** file.

Part 7: Cleanup

1. Restore your pre-DNS configuration on all hosts in your domain by running **netfiles.sh**:

```
master# /labs/netfiles.sh -r NEW
 slave# /labs/netfiles.sh -r NEW
client# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

13–14. LAB: Configuring and Securing ARPA/Berkeley Services

Directions

This lab offers an opportunity to configure, use, and troubleshoot the ARPA/Berkeley service configuration on your machine. For a portion of the lab, you will need to work with a partner. Choose a partner, and decide which machine will be the internet service "server" during the experiments that follow, and which will be the "client".

Note that the "server" and "client" roles assigned in this lab are relatively arbitrary. Most HP-UX machines are configured to provide both client and server functionality.

```
Server's host name: _____________________

Client's host name: _____________________
```

Part 1: Basic ARPA/Berkeley Service Configuration

1. (server and client) The "InternetSrvcs" product must be installed on every machine that wishes to use or provide ARPA/Berkeley services. Check to ensure that this product is installed on your system.

   Answer

   ```
   # swlist -l product InternetSrvcs
   ```

2. (server) The server's **inetd** daemon must be running in order for clients to have access to any of the internet services. Use **ps -e** to check to ensure that the **inetd** daemon is running on your server.

   Answer

   ```
   # ps -e | grep inetd
   ```

3. (server and client) Which script starts **inetd** during the boot process? At which run level does **inetd** start?

   Answer

   inetd is started by **/sbin/init.d/inetd** at run level 2, and is killed by the same script at run level 1.

4. (server) Look at **/etc/inetd.conf** and **/etc/services** to determine which Internet services are configured on your server, then complete the table below:

   ```
   Service    Enabled?    Port#
   -------    --------    -----
   telnet
   ftp
   login
   tftp
   bootps
   ```


© 2003 Hewlett-Packard Development Company, L.P.

5. Do you currently have server processes running for these services? Explain.

Answer

```
# more /etc/inetd.conf
# more /etc/services
```

The list of services enabled may vary from machine to machine, depending on the contents of **/etc/inetd.conf**. Services that are commented out are not available.

The port numbers for the services may be found in the second field of the **/etc/services** file.

Most likely, there are no server processes running for any of the listed services. Server processes for these services are only started on an as-needed basis.

6. (server) Ensure that the services in **inetd.conf** that appear to be enabled actually are enabled. Use **netstat -a** to check the status of each of the enabled services and ports you listed in the table above.

Answer

```
# netstat -a
```

netstat -a lists the status of all configured ports. Unless the services are currently in use, all ports associated with the services listed in the table should all be in a "LISTEN" state.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Securing the Internet Services

1. (server) The **inetd.conf** file allows you to enable or disable an Internet service for *all* clients. If, however, you wish to allow/prevent *specific* client(s) access to a service, you must use the **/var/adm/inetd.sec** file.

   Configure your **/var/adm/inetd.sec** file such that only the hosts in your row (including your partner) have **telnet** access. Add another line to ensure that all your classmates except your partner can **ftp** to your machine.

   Answer

   ```
   vi /var/adm/inetd.sec
       telnet allow 128.1.1.1-4  # actual IP addresses will vary
       ftp deny 128.1.1.2        # actual IP addresses will vary
   ```

2. (client) See if your server's configurations so far have succeeded. What messages do you see when you attempt to **telnet** or **ftp** to the server?

   Answer

   telnet succeeds.

   ftp fails with the message: "**Service not available**."

3. (server) What do you have to do to enable **inetd** logging? Make it so.

   Answer

   ```
   # vi /etc/rc.config.d/netdaemons
         export INETD_ARGS="-l"

   # /sbin/init.d/inetd stop
   # /sbin/init.d/inetd start
   ```

4. (client) See if the logging feature works. From the client, **telnet** to the server, do an **ls**, then immediately exit. Then attempt to **ftp** to the server (this should fail). Move on to the next question to see what was recorded in the **inetd** log.

   Answer

   ```
   # ftp server     # server host name will vary
   # telnet server  # server host name will vary
   ```

5. (server) How much detail is recorded in the **inetd** log? On the server, do a **more** on the file where ARPA/Berkeley service requests are logged.

   - Does **inetd** log the name of the service requested?
   - Does **inetd** log the host name of the requesting client?
   - Does **inetd** log the username of the user making **telnet** requests?
   - Does **inetd** log the commands executed during the **telnet** session?
   - Does **inetd** log denied requests for Internet service?

© 2003 Hewlett-Packard Development Company, L.P.

Answer

Looking at **`/var/adm/syslog/syslog.log`**, you should see that:

Yes, the service name is recorded.

Yes, the requesting client host name is recorded.

No, username requesting a **`telnet`** connection is not recorded.

No, commands executed during a **`telnet`** session are not recorded.

Yes, denied service requests are recorded.

© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Experimenting with ARPA/Berkeley Service Connections

The goal of this part of the lab is to determine what happens when a client process connects to a server providing ARPA/Berkeley services. More specifically, we will be experimenting with the "**telnet**" service.

1. (client and server) First, check to see which daemons and processes are already running on the server and client:

```
client# ps -e | grep telnet
server# ps -e | grep telnet
```

Answer

There should not be any **telnet** sessions running at this point.

2. (client and server) Establish a **telnet** session from the client to the server, and look at the process table to determine which processes were started as a result.

```
client# telnet server
client# ps -e | grep telnet
server# ps -e | grep telnet
```

Which **telnet** related processes are running on the client now? Which **telnet** related processes are running on the server now?

Answer

On the client, there should be a **telnet** process.
On the server, there should be a **telnetd** process.

3. (client and server) Can multiple clients **telnet** to the server simultaneously? Try it. On the client side, open another window and initiate another **telnet** connection to the server. Check to see which **telnet** related processes are running on the server and client:

```
client# ps -e | grep telnet
server# ps -e | grep telnet
```

How many **telnetd** server processes are running on the server? How many **telnet** processes are running on the client? Explain.

Answer

There should be two **telnet** processes on the client, and two **telnet** server processes on the server.

Every **telnet** service request first goes to the server's **inetd** daemon, at which point **inetd** starts the appropriate server process to manage interaction with the requesting client.

You will have multiple **telnetd** server processes running on your machine if there are multiple simultaneously connected clients.

© 2003 Hewlett-Packard Development Company, L.P.

4. (client and server) Take a look at the ports that are being used by your **telnet** processes:

```
client# netstat -a | grep telnet
server# netstat -a | grep telnet
```

How many **telnet** connections are ESTABLISHED? What process do you suppose is monitoring the port in the LISTEN state? Do the client side **telnet** processes share a port or use different ports? Which well-known port number are the **telnetd** daemons on the server sharing?

Answer

Two connections should be established.

inetd is LISTENing on port number 23 for additional **telnet** requests.

On the client side, the **telnet** processes each have a separate port.

On the server side, however, all the **telnet** daemons receive data on port 23.

5. (client) Close your **telnet** connections to the server.

© 2003 Hewlett-Packard Development Company, L.P.

Part 4: Experimenting with ARPA/Berkeley Services

1. (client) What happens if the server's **inetd** daemon is down when a client attempts to connect? Try it, then explain the result.

```
server# inetd -k              # kill the server's inetd
client# telnet server         # can the client still connect?
server# inetd                 # restart the server's inetd
```

Answer

The connection fails. Clients cannot connect until the server's **inetd** daemon returns.

2. (client and server) What happens if the server's **inetd** daemon goes down AFTER a session has been established -- does the existing connection remain, or are all client connections immediately terminated? Try it, then explain the result.

```
client# telnet server    # establish a connection to the server
server# inetd -k              # kill the server's inetd.
server# ps -e | grep telnetd  # does the telnet daemon remain?
```

Answer

Existing connections remain, even if **inetd** is killed. After the initial connection, **inetd** is no longer involved in the client - server communication.

3. (client and server) What happens if the server's **telnetd** server process is killed while a client is connected? Try it.

```
server# ps -e | grep telnetd  # find the server process's PID
server# kill _____            # kill telnetd's PID
```

Does the client **telnet** process exist after the server's **telnetd** daemon is killed?

Restart **inetd** on the server before proceeding to the next question.

```
# inetd
```

Answer

Killing the **telnetd** process on the server severs the connection. The client **telnet** process dies as a result.

4. (client) Must the client be running **inetd** in order to establish connections to a server? Try it, and explain the result.

```
client# inetd -k              # kill the client's inetd
client# telnet server         # can the client still telnet out?
client# inetd                 # restart the client's inetd
```

Answer

Even if the client's **inetd** process dies, the client should still be able to **telnet** out.

© 2003 Hewlett-Packard Development Company, L.P.

Part 5: Experimenting with Host and User Equivalency

1. **(server)**
   Configure host equivalency for all the hosts in your row, including your client.

   Answer

   ```
   # vi /etc/hosts.equiv          # list all hosts in your row, one
                                    per line, by host name.
   ```

2. **(client)**
   While logged in as root, use **rlogin** to log into the server. What happens? Why?
   Exit out of your **rlogin** session before proceeding to the next question.

   Answer

   ```
   # rlogin server
   ```

   You should still be prompted for a password. Remember, host equivalency does not apply to the root account.

   ```
   # exit
   ```

3. **(client)**
   Use the **su** command to switch your user ID to user1. Then try **rlogin** again. What happens? Why?

   Answer

   ```
   # su - user1
   # rlogin server
   ```

   This should work. **/etc/hosts.equiv** on the server grants host equivalency to users on the client.

4. **(server)**
   What can you do on the server to enable root on the clients password free access to your machine? Make it so.

   Answer

   ```
   # vi ~root/.rhosts             # add the client's host name to the file
   ```

5. **(client)**
   Terminate the **rlogin** and **su** sessions you stated previously. Ensure that you are back to the "root" userid. Then see if you can **rlogin** to the server without a password

   Answer

   ```
   # exit
   # exit
   # rlogin server               # should work!
   ```


© 2003 Hewlett-Packard Development Company, L.P.

6. **(server)**
Remove **/etc/hosts.equiv** and **~root/.rhosts**.

Answer

```
# rm /etc/hosts.equiv ~root/.rhosts
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 6: (Optional) Troubleshooting Problems with the Internet Services

In the exercise that follows, you will work with your partner to corrupt, then fix the internet service configuration on the server machine that you chose at the beginning of this lab.

The list below suggests several different ways to corrupt the internet service configuration on your "server" machine. Take turns being the "corrupter" and the "troubleshooter."

The "corrupter" should perform any one of the corruption techniques from the list below on the "server" machine. It is the duty of the "troubleshooter," then to do whatever is necessary on the server to enable the client to successfully **telnet** to the server.

Try the exercise several times, alternating roles as "corrupter" and "troubleshooter."

Before starting the exercise, shutdown CDE:

```
/sbin/init.d/dtlogin.rc stop
```

Eight Ways to Corrupt an Internet Service Server

1. Kill the **inetd** daemon with **inetd -k**
2. Comment out the **telnet** line in **/etc/inetd.conf** and restart **inetd**.
3. Comment out the **telnet** line in **/etc/services** and restart **inetd**.
4. Take down the server's LAN card with **ifconfig lan0 down**.
5. Change the server's IP address with **ifconfig lan0 254.254.254.254**.
6. Detach the LAN cable on the server.
7. Change the client's network entry in the server's routing table.
8. Deny the client **telnet** access via **/var/adm/inetd.sec**.

Part 7: Cleanup

Before moving on to the next chapter, restore your network configuration to the state it was in before this lab.

```
# /labs/netfiles.sh -r NEW
```


© 2003 Hewlett-Packard Development Company, L.P.

13–15. REVIEW QUESTIONS: Configuring and Securing ARPA/Berkeley Services

Directions

Answer the following questions.

1. What is the difference between a daemon and a server process?

   Answer

   A daemon is a software process that runs continuously (in the background) and provides services upon request. A server process runs one time, when called by a daemon, and then stops.

2. List some Internet Services daemons and server processes.

   Answer

   Daemons: **inetd**, **rwhod**, **sendmail**, **named**
   Server processes: **telnetd**, **ftpd**, **rlogind**, **rexecd**, **remshd**

3. What does **inetd** do? What is the advantage in its functioning?

   Answer

   inetd is a "superdaemon"; it is responsible for invoking other Internet servers when they are needed. By allowing this one daemon to invoke many servers, the system load is reduced. (The alternative would be to have one daemon for each of the services, which would significantly increase the load.)

4. What is the name of the **inetd** configuration file?

   Answer

   ```
   /etc/inetd.conf
   ```

5. What command do you use after modifying the configuration file?

   Answer

   ```
   # inetd -c
   ```

6. What is a port? What file associates port numbers with a service name?

   Answer

   A port is an address within a host that is used to differentiate between multiple sockets with the same Internet address. Ports are identified by port numbers. (A socket address

© 2003 Hewlett-Packard Development Company, L.P.

consists of the Internet address plus the port number.)

The **/etc/services** file associates a port number with a service name. These ports are called well-known ports.

7. List at least four security features of the Internet Services.

   Answer

   - **/etc/hosts.equiv**
   - **$HOME/.rhosts**
   - **/etc/ftpusers**
   - **/etc/inetd.conf**
   - **/var/adm/inetd.sec**
   - **inetd** logging

8. Which server processes use the **/etc/hosts.equiv** and **$HOME/.rhosts** files?

   Answer

   The servers **remshd** and **rlogind** use these files, if the files are present.

9. Are the **/etc/hosts.equiv** and **$HOME/.rhosts** files optional for using the Berkeley Services? Explain your Answer.

   Answer

   The Answer is yes and no. These files are optional, they are used if they are present. You need to configure bypass password security (user or system equivalency) if a remote user is able to access to one of your password secured user accounts with **rcp** or **remsh**.

10. What is the name and what are the features of the security file that **ftpd** uses?

    Answer

    The **/etc/ftpusers** file denies remote users to access the specified users with **ftp**.

11. What is an anonymous FTP?

    Answer

    A public user account without password security that allows a user to copy files with **ftp** from or to a remote system . A **chroot()** is done to the anonymous FTP user's HOME directory.

© 2003 Hewlett-Packard Development Company, L.P.

12. What is the security feature of **/var/adm/inetd.sec**?

    Answer

    It allows or denies certain services that are administrated by **inetd** for specific hosts or networks.

13. What is wrong in the following **inetd.sec** example?

    ```
    rlogin allow 192.6.1
    ```

    Answer

    This will not work. The official service name (see **/etc/services**) is **login**, not **rlogin**.

14. If **inetd** logging is enabled, which file contains the logging output?

    Answer

    The log file of **syslogd** is **/var/adm/syslog/syslog.log**.

© 2003 Hewlett-Packard Development Company, L.P.

14–9. LAB: Managing a bootp/tftp Server

Part 1: Basic bootp/tftp Configuration

1. The **bootp/tftp** services are bundled in the InternetSrvcs product. Ensure the InternetSrvcs product is installed on your machine.

```
# swlist -l product InternetSrvcs
```

2. Ensure that the **bootps** and **tftp** services are both enabled in **/etc/inetd.conf** and the **/etc/services** file.

```
# more /etc/inetd.conf  # Are tftp and bootps commented in?
# more /etc/services    # Are tftp and bootps commented in?
# inetd -c              # Reread inetd.conf if necessary
```

3. Files available for download via **tftp** are often stored in a **tftp** home directory. If you don't already have a **tftp** account in your **/etc/passwd** file, create it from the SAM Network services screen.

```
SAM --> Networking and communications --> Network services
        (select TFTP)
        Actions --> Enable
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring `bootp/tftp` Service for a Network Printer

1. Using **`hppi`**, create a **`bootptab`** entry for a network printer. Use the hardware address, **`IP address`**, host name, subnet mask, and default router address provided by your instructor. Use your classroom's room name or number as the **printer location**, and your own name as the printer **contact**.

```
# /opt/hpnpl/bin/hppi
  -> (2) JetDirect Configuration
  -> (1) Create printer configuration in BOOTP/TFTP database
```

(Answer the questions that follow according to instructor's directions.)

NOTE:	The **`hppi`** utility may complain that a "network port is being used by **`rbootd`**." Despite this warning, **`hppi`** should successfully configure the printer. The warning can be ignored.

2. Check the **`/etc/bootptab`** file for changes made by **`hppi`**. Name three pieces of information defined in the printer's new entry in **`bootptab`**.

Answer

The following are a few of the most common fields found in the **`/etc/bootptab`** file:

```
:ht=    #Network interface card type (ether, ieee, etc.)
:ha=    #MAC address
:hn:    #Should BOOTP provide the printer a host name?
:ip=    #IP address
:sm=    #Subnet mask
```

3. At this point your machine is ready to service **`bootp`** requests from the network printer you configured.

4. Now remove the new printer **`bootp`** configuration from your machine using **`hppi`**.

```
# /opt/hpnpl/bin/hppi
  -> (2) JetDirect Configuration
  -> (2) Remove printer configuration from BOOTP/TFTP database
```


© 2003 Hewlett-Packard Development Company, L.P.

Part 3: Configuring bootp/tftp Service for an X terminal

1. Hosts that manage X terminals must have the "Enware" product installed. Check to ensure that your machine has "Enware" installed.

```
# swlist -l product ENWARE
```

2. Some X terminals download the X server software via **tftp**, while others use **nfs**. Ensure that your host is configured as an NFS server, and that the **/opt/hpxt** directory is included in your **/etc/exports** file.

```
# ps -e | grep nfsd      # nfs server process should be running!
# exportfs               # /opt/hpxt should be exported
```

3. Now configure an X terminal entry in your **/etc/bootptab** file using **xtadm**. Use the MAC, IP, host name, subnet mask and default router address suggested by your instructor.

```
# /opt/hpxt/enware/bin/xtadm
       --> (1) Add an X station
 (Answer q's that follow using addresses provided by instructor)
```

4. Look at the **/etc/bootptab** entry that was created for your X terminal. List three pieces of information defined for your X terminal in the new **bootptab** entry.

```
:ht=    #Network interface card type (ether, ieee, etc.)
:ha=    #MAC address
:hn:    #Should BOOTP provide the X terminal a host name?
:ip=    #IP address
:sm=    #Subnet mask
(others possible, too)
```

5. X terminals download a configuration file from the **bootp/tftp** server. These configuration files are normally stored in **/opt/hpxt/enware/xthome/config/**. Your new X terminal should have an entry in this directory. Search through the file for the lines labeled "Startup Session" and "XDMCP Host." These two lines determine the session server that your new X terminal will connect to.

   By default, the X terminal will connect to the **bootp/tftp** server as the session server as well. You can specify another session server by adding the following lines anywhere in the configuration file:

```
Startup Session = XDMCP Direct
XDMCP Host      = host name_of_session_server
```

   Configure your new X terminal to connect to the session server named by your instructor.

© 2003 Hewlett-Packard Development Company, L.P.

6. Now your host is ready to provide **bootp/tftp** server to your newly defined X terminal. If the X terminal you defined is available in the classroom, reboot it to see if your configuration was successful.

7. Remove the **bootp** entry you added for your X terminal using **xtadm**.

```
/opt/hpxt/enware/bin/xtadm
          -->(2) Remove an X Station
```

(Answer the questions that follow)

© 2003 Hewlett-Packard Development Company, L.P.

15–11. LAB: Introduction to NTP

Directions

Your instructor will assign you to work with a team of your classmates to configure an NTP server, and one or more NTP clients. Record the host names and chosen roles of your teammates' machines below.

```
NTP server: _____________

NTP client: _____________

NTP client: _____________
```

Record the commands you use to complete the steps below, and Answer all questions.

Part 1: Configuring an NTP Server

The steps below should only be configured on the host you have chosen to be the NTP server. Do not start configuring the NTP clients until the server configuration is complete.

Since you probably do not have access to a radio clock in the classroom, use the NTP server's internal system clock as the authoritative time source for your team.

1. Set the local clock forward 2 minutes so the clients can actually see a clock "step" after enabling NTP.

   ```
   date MMDDhhmm
   xclock -update 1 &
   ```

2. Add a server line to the end of the **/etc/ntp.conf** file defining the local clock as the only time source. Since the internal system clock is not likely to be accurate, set the stratum level of this time source to 10.

   ```
   # vi /etc/ntp.conf
         server 127.127.1.1
         fudge 127.127.1.1 stratum 10
   ```

3. Modify the **/etc/rc.config.d/netdaemons** file to enable XNTPD on the server. Do not specify an NTP date server.

   ```
   # vi /etc/rc.config.d/netdaemons
         NTPDATE_SERVER=
         XNTPD=1
         XNTPD_ARGS=
   ```

4. Run the NTP startup script to start the **xntpd** daemon.

   ```
   # /sbin/init.d/xntpd start
   ```


© 2003 Hewlett-Packard Development Company, L.P.

5. After **xntpd** starts, it takes 5 poll cycles (320 seconds) to establish the appropriate peer and server relationships. Wait 5 minutes before proceeding on to the next question.

6. Is the **xntpd** daemon running? Are there any NTP errors in the syslog?

   ```
   # ps -e | grep xntpd

   # tail /var/adm/syslog/syslog.log
   ```

   If all is well, the daemon should be running, and there should not be any XNTPD "ERROR"s in the syslog.

7. Does **ntpq -p** suggest that the correct association has been formed? What stratum level did NTP assign to your local clock?

   ```
   # ntpq -p
   ```

   There should be one line in the **ntpq -p** output showing that the local clock is being used as a stratum 10 time source.

© 2003 Hewlett-Packard Development Company, L.P.

Part 2: Configuring an NTP Client

Do not start this procedure until you confirm that your NTP server is fully functional. The steps below should only be performed on the host(s) you have chosen as NTP clients.

1. Add appropriate server and driftfile lines to your **/etc/ntp.conf** file to poll the NTP server created in the previous portion of the exercise.

   ```
   vi /etc/ntp.conf
   server 128.1.1.1          # assume 128.1.1.1 is the NTP srvr IP
   driftfile /etc/ntp.drift
   ```

   You may use the server's hostname rather than the IP if you wish.

   Note: **xntp** must be able to write to the directory where the drift file is located.

2. Modify the **/etc/rc.config.d/netdaemons** file to enable **xntpd**. Also define your NTP server to be the NTPDATE_SERVER.

   ```
   # vi /etc/rc.config.d/netdaemons
         NTPDATE_SERVER=128.1.1.1   # Assume 128.1.1.1 is the NTP srvr IP
         XNTPD=1
         XNTPD_ARGS=
   ```

   Here again, you may use the server's host name in place of the IP if you wish.

3. Run the NTP startup script on the client to start the NTP daemon. Note the output as **ntpdate** steps the system clock.

   ```
   # /sbin/init.d/xntpd start
   ```

4. Check to ensure that your client formed the proper association by running **ntpq -p**.

   ```
   # ntpq -p
   ```

5. Compare the time on your client against the time on the NTP server. Do they appear to be synchronized at this point?

   Answer

   Execute the **date** command on both machines.
   They should agree.

© 2003 Hewlett-Packard Development Company, L.P.

16–7. LAB: Creating and Managing an SD-UX Depot

Directions

Create a software depot and install software products from this new depot.

Perform all the tasks suggested below. Record the commands you use, and Answer all questions.

1. What is an SD depot?

   Answer

   A depot is a repository of software products organized such that the SD can use it as a software source. A depot can exist as a directory tree on a HP-UX file system (the default is **/var/spool/sw**), a directory tree on CD-ROM media, or a **tar** archive on a serial media.

2. Create an SD-UX depot server, choose any path you like for your directory depot. Load only one product.

   Answer

   If the source is a DDS tape or another SD server, create your depot with **swcopy**. If you have a CD-ROM as source, mount it and register it with **swreg**. See the student notes.

3. List the contents of your depot.

   Answer

   Execute

   ```
   swlist -d @ /depot_path | more
   ```

4. List a remote depot. Ask for a depot path your neighbor team and look at the manual page to get the syntax.

   Answer

   Execute

   ```
   swlist -d @ hostname:/depot_path | more
   ```

5. Install a product, using your own depot or a remote depot.

   Answer

   Use **swinstall**.

© 2003 Hewlett-Packard Development Company, L.P.

6. Check the installation of the product with **swlist** and sw**v**erify.

Answer

Execute

```
# swlist product
# swverify product
```


© 2003 Hewlett-Packard Development Company, L.P.