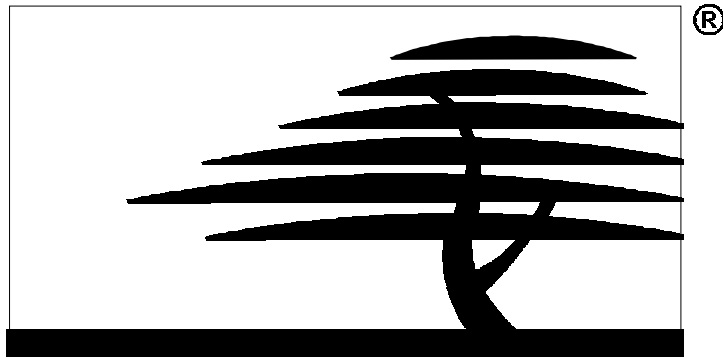




SCO

UnixWare[®] 7 Network Administration

Student Guide

Course Release: 2.0

For Software: UnixWare[®] 7.1

© 1987–1999 The Santa Cruz Operation, Inc. All rights reserved.

This publication is protected under copyright laws and international treaties.

Information in this document is subject to change without notice and does not represent a commitment on the part of The Santa Cruz Operation, Inc.

SCO, The Santa Cruz Operation, the SCO logos, VP/ix, SCO OpenServer, UnixWare, dbXtra, Panner, SCO Global Access, SCO MPX, SCO Doctor, SCO Doctor for Networks, SCO Doctor Lite, SCO Premier Motif, SCO Visual Tcl, SCO VisionFS, SCO ToolWare, SCO TermLite, SCO Vision 97, SCO CIFS Bridge, SCO WebTop, The Internet Way of Computing, and IWoC are trademarks or registered trademarks of The Santa Cruz Operation, Inc. in the USA and other countries. IXI, X.desktop, Deskworks, Wintif, IXI Panorama, Deskterm, and IXI Desktop are trademarks or registered trademarks of IXI Limited, a subsidiary of The Santa Cruz Operation, Inc. Visionware, Devkit.Vision, Esprit, Kodon, Super.Vision, Visionware Super.Vision, Term.Vision, Visionware Direction, the Visionware logo, Visionware SQL-Retriever, and XVision are registered trademarks of Visionware Limited, a subsidiary of The Santa Cruz Operation, Inc. SuperVision, TermVision, X-Visionware, SQL-Retriever, and Vision Builder are trademarks of Visionware Limited. UNIX is a registered trademark of The Open Group in the United States and other countries. Cheyenne and ARCServe are registered trademarks of Cheyenne Software, Inc. Netscape, Netscape Navigator, Netscape Communications Server, Netscape Commerce Server, Netscape Proxy Server, Netscape FastTrack Server, Netscape Enterprise Server, Netscape SuiteSpot, Netscape Catalog Server, Netscape News Server, Netscape Mail Server, and Netscape Navigator Gold are trademarks of Netscape Communications Corporation. NFS was developed by Legent Corporation (formerly Lachman Associates, Inc.) based on LACHMAN SYSTEM V NFS. LACHMAN is a trademark of Legent Corporation. NFS is a trademark of Sun Microsystems, Inc. TCP/IP was developed by Legent Corporation (formerly Lachman Associates, Inc.) based on LACHMAN SYSTEM V STREAMS TCP, a joint development of Lachman Associates and Convergent Technologies. MPX was developed by Corollary, Inc. VP/ix is a product developed and licensed by Phoenix Technologies, Ltd/INTERACTIVE Systems Corporation. XRemote is a registered trademark of Network Computing Devices, Inc. Oracle is a registered trademark of Oracle Corporation, Redwood City, California. Java is a trademark of Sun Microsystems, Inc. All other brand and product names are or may be trademarks of, and are used to identify products or services of, their respective owners.

The SCO software that accompanies this publication is commercial computer software and, together with any related documentation, is subject to the restrictions on US Government use as set forth below. If this procurement is for a DOD agency, the following DFAR Restricted Rights Legend applies:

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the US Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013. Contractor/Manufacturer is The Santa Cruz Operation, Inc., 400 Encinal Street, Santa Cruz, CA 95060.

If this procurement is for a civilian government agency, this FAR Restricted Rights Legend applies:

RESTRICTED RIGHTS LEGEND: This computer software is submitted with restricted rights under Government Contract No. _____ (and Subcontract No. _____, if appropriate). It may not be used, reproduced, or disclosed by the Government except as provided in paragraph (g)(3)(i) of FAR Clause 52.227-14 alt III or as otherwise expressly stated in the contract. Contractor/Manufacturer is The Santa Cruz Operation, Inc., 400 Encinal Street, Santa Cruz, CA 95060.

The copyrighted software that accompanies this publication is licensed to the End User only for use in strict accordance with the End User License Agreement, which should be read carefully before commencing use of the software.



The SCO® Authorized Education class you are about to take may form part of the recommended preparation for those seeking the SCO ACE™ certification.

The SCO ACE certification program uses a series of exams to measure your ability to support and administer SCO installations with the SCO recommended skills that maximize system performance. As a certified SCO ACE graduate, you earn all of the respect that the industry accords to each certified member of the extended SCO support team. When professionals in the computer industry see your SCO ACE certification, they know that you have passed demanding SCO testing requirements. This certification is the foundation for mastering real-world UNIX® systems, including those from other vendors.

Important!

When registering for any SCO ACE exam at a Sylvan Prometric™ testing center, please quote the course release number on the front of the ***Student Guide*** that the exam covers. This number corresponds to the version number of the SCO ACE exam for that course.

Ask your SCO Authorized Instructor, or call SCO for more information about SCO Certification.

Table of Contents

Lesson 1: Course Introduction	1-1
Introduction.....	1-2
Course goal and purpose.....	1-2
Target audience	1-2
Student Materials	1-3
The <i>Student Guide</i>	1-3
SCO documentation.....	1-3
Typographical Conventions	1-4
Sample exam items	1-6
 Lesson 2: TCP/IP Concepts	 2-1
Hardware Addresses	2-4
Software Addresses.....	2-9
IP addresses.....	2-9
Netmasks.....	2-15
Network addresses	2-16
Broadcast addresses	2-17
Available host addresses	2-20
Multicast addresses	2-21
Network-Joining Devices	2-23
Host Names and IP Addresses	2-25
System name	2-26
Domain names	2-27
The <i>/etc/hosts</i> file.....	2-29
Translating IP Addresses to Hardware Addresses	2-36
Address resolution protocol (ARP).....	2-36
The ARP cache and the arp (1Mtcp) command.....	2-37
Protocols and Ports	2-38
Clients and servers	2-38
Ports	2-39
Transport protocols	2-40
The TCP/IP Protocol Stack.....	2-43
TCP/IP layers.....	2-45
Transmission between the layers	2-47
An example transmission.....	2-48
LAN Hardware	2-50
Ethernet.....	2-50
IBM Token Ring	2-53
FDDI	2-55
ATM	2-55
The Future of the Internet Protocol (IPv6)	2-56

Lesson Summary	2-57
Relevant Commands, Files, and Syntax	2-59
Answers to Review Questions	2-60

Lesson 3: Configuring a Network Adapter	3-1
--	------------

Configuring Network Interfaces	3-4
Network hardware.....	3-5
Network adapter drivers.....	3-7
Testing TCP/IP Connectivity	3-12
The netstat (1Mtcp) command.....	3-13
The ping (1Mtcp) command.....	3-14
The <i>loopback</i> interface.....	3-15
Verifying network hardware	3-17
Configuring Network Interface Components	3-20
Adding protocols.....	3-20
Modifying and removing components	3-21
Configuring the /etc/hosts File.....	3-23
Configuring Network Adapter Failover.....	3-24
Lesson Summary	3-27
Lab: Configuring a Network Node	3-29
Configuring a network adapter	3-29
Relevant Commands and Files	3-32
Answers to Review Questions	3-33

Lesson 4: Subnets	4-1
--------------------------	------------

Planning a Subnet Topology	4-4
Increasing the number of network nodes.....	4-7
Joining multiple LANs.....	4-7
Increased network traffic	4-9
Creating Subnets	4-11
Dividing up the network	4-11
Class A subnets.....	4-15
Class B subnets	4-23
Class C subnets	4-30
Unusual subnet masks.....	4-37
Lesson Summary	4-38
Relevant Commands and Files	4-40
Answers to Review Questions	4-41

Lesson 5: Routing	5-1
Configuring a System as a Router	5-5
Installing additional network adapters	5-7
Kernel parameters	5-9
Configuring Routes	5-11
Direct and indirect routing	5-11
Routing table	5-13
The netstat(1Mtcp) command	5-15
Updating routes	5-17
Manual updates and route(1Mtcp)	5-19
Default routes	5-21
Dynamic Routing Table Updates and Routing Daemons	5-25
routed(1Mtcp)	5-27
gated(1Mtcp)	5-33
mrouted(1Mtcp)	5-35
Troubleshooting Routes	5-37
The ping(1Mtcp) command	5-37
The tracert(1Mtcp) command	5-39
The rtquery(1Mtcp) command	5-41
The ripquery(1Mtcp) command	5-41
Lesson Summary	5-42
Lab: Configuring a Router	5-43
Basic router setup	5-43
The ipforwarding parameter	5-45
Relevant Commands and Files	5-47
Answers to Review Questions	5-49
 Lesson 6: Configuring WAN Connections	 6-1
WAN Interface Types	6-4
Modems	6-5
ISDN	6-11
Other digital subscriber line technologies	6-13
Leased lines	6-15
Frame relay	6-17
Cable modems	6-17
WAN Protocols	6-19
Serial Line Internet Protocol (SLIP)	6-19
Point-to-Point Protocol (PPP)	6-21
Configuring PPP Connections	6-25
Configuring devices for WAN connections	6-25
Creating outgoing PPP connections	6-31
Using an outgoing PPP connection	6-41
Creating incoming PPP connections	6-43
Troubleshooting PPP	6-49
Manually testing a connection	6-49

Using PPP Logs	6-51
Check that the PPP interface exists.....	6-52
Verify network connection between hosts	6-53
Check routing.....	6-53
Reset PPP	6-54
Lesson Summary.....	6-55
Lab: Configuring PPP Connections	6-56
Disabling LAN Access	6-56
Configuring a Modem for Outgoing Calls.....	6-56
Configuring an Outgoing PPP Connection.....	6-59
Configuring a Modem for Incoming Calls	6-62
Configuring an Incoming PPP Connection.....	6-63
Configuring an Outgoing Connection Revisited	6-65
Relevant Commands and Files	6-66
Answers to Review Questions	6-68

Lesson 7: TCP/IP Administration Topics	7-1
---	------------

TCP/IP Start-up.....	7-4
Configuring protocol-independent STREAMS modules.....	7-5
Configuring basic Internet services	7-7
Installing packet filtering.....	7-8
Initializing PPP, SNMP, and RPC	7-9
Daemon Startup	7-10
The <i>config(4tcp)</i> file	7-13
Running programs from <i>/etc/inet/config</i>	7-14
The inetd (1Mtcp) Super Daemon.....	7-17
Configuring inetd through the <i>inetd.conf</i> file.....	7-18
Enabling and disabling daemons	7-19
Adding new services	7-19
TCP wrappers and tcpd (1Mtcp)	7-21
Using telnet to test daemons	7-22
Configuring Trusted Access.....	7-25
Configuring trusted host access	7-27
Configuring trusted user access	7-29
The trusted access sequence	7-33
Overview of user equivalence.....	7-37
Changing an account's UID and login name	7-42
Lesson Summary.....	7-43
Relevant Commands and Files	7-44
Answers to Review Questions	7-46

Lesson 8: Domain Name System (DNS) Configuration	8-1
DNS Operation	8-5
DNS server types	8-9
Configuring DNS Name Servers	8-17
How named works	8-19
The <i>named.conf</i> file	8-21
The <i>db.somedom.com</i> file	8-33
The <i>db.50.100.200</i> file	8-33
The <i>db.local</i> file	8-35
The <i>db.cache</i> file	8-35
DNS Records	8-37
The DNS Manager	8-51
Controlling the name server	8-57
Command line control of the name server	8-59
Querying a name server	8-61
The nslookup (1Mtcp) command	8-61
Configuring DNS Clients	8-63
DNS client configuration	8-63
Lesson Summary	8-66
Lab: Configuring and Testing Name Servers and Clients	8-68
Configuring servers	8-68
Testing your server	8-71
Configuring a client	8-72
Relevant Commands and Files	8-74
Answers to Review Questions	8-76
 Lesson 9: Configuring Internet Services	 9-1
Virtual Domains	9-5
Creating virtual domains	9-7
Virtual domain services	9-9
DNS and virtual domains	9-10
Configuring and Using Web Services	9-11
Starting and stopping Netscape HTTP servers	9-12
Configuring the Netscape FastTrack Server	9-15
Running the Netscape Administration Server	9-17
Configuring remote administration	9-21
Web Services for Virtual Domains	9-27
Adding a virtual domain web server	9-27
Modifying the virtual domain home page	9-33
Configuring an FTP Server	9-37
Enabling and disabling FTP	9-37
Configuring anonymous FTP	9-39
Anonymous FTP for Virtual Domains	9-43
Configuring anonymous FTP for a virtual domain	9-43
Configuring anonymous FTP directories	9-45

Other FTP configurations.....	9-47
Limiting access using classes	9-47
Other FTP configurations.....	9-53
Flags for ftpd (1Mtcp).....	9-61
Configuring Time Synchronization	9-63
Lesson Summary.....	9-67
Relevant Commands and Files	9-68
Answers to Review Questions	9-71

Lesson 10: Network Security	10-1
------------------------------------	-------------

TCP Wrappers.....	10-5
Configuring inetd to use TCP wrappers.....	10-6
The <i>hosts.allow</i> and <i>hosts.deny</i> files.....	10-6
Firewall Basics.....	10-11
Packet filters	10-13
Proxy Servers.....	10-25
Using a proxy server	10-26
Packet Security	10-27
Packet sniffing	10-29
The Secure Socket Layer (SSL).....	10-31
Lesson Summary.....	10-32
Relevant Commands and Files	10-34
Answers to Review Questions	10-35

Lesson 11: Windows Connectivity	11-1
--	-------------

File and Printer Sharing	11-5
NFS	11-6
NetWare Services.....	11-7
VisionFS.....	11-7
Installing and Configuring VisionFS	11-9
VisionFS passwords.....	11-9
Installing VisionFS	11-10
Running the password wizard.....	11-11
Adding users and changing passwords	11-12
Windows network configuration.....	11-15
VisionFS administration overview.....	11-19
Sharing files and directories with VisionFS.....	11-23
Sharing printers using VisionFS	11-31
Advanced File and Print Server	11-39
Remote Application Access.....	11-41
Terminal emulation	11-42
X Window servers.....	11-49
Custom client agents.....	11-50
Webtop	11-52
Remote Boot Configuration.....	11-53

Configuring an Address Allocation Server	11-55
Configuring a DHCP server	11-59
Lesson Summary	11-76
Relevant Commands, Directories, Shares, and Packages	11-78
Answers to Review Questions	11-79
 Lesson 12: Mail Configuration	 12-1
Overview of E-mail	12-5
The Mail User Agent (MUA)	12-6
The Mail Transport Agent (MTA)	12-7
The mail channels and delivery agents	12-8
POP and IMAP servers	12-10
Configuring sendmail	12-12
Default sendmail configuration	12-12
Using the Mail Manager to modify sendmail	12-15
Configuring Folders	12-17
Configuring Aliases	12-19
Modifying the default alias file	12-19
Alias subscription	12-22
Mail Delivery Channels	12-22
Configuring Sendmail Manually	12-23
The <i>sendmail.cf</i> file	12-23
Alias files	12-24
User-configurable mail options	12-25
<i>.forward</i> files	12-25
Extended absence notification	12-27
Maildelivery Files	12-28
Mail and DNS	12-29
Client-Server Mail (POP and IMAP)	12-31
Lesson Summary	12-32
Relevant Commands and Files	12-33
Answers to Review Questions	12-34

1

Course Introduction

1.1 Introduction

This **UnixWare® 7 Network Administration** course is designed for technically-oriented individuals who need to establish and maintain the network connectivity of their UnixWare 7 operating system with other networks, including the Internet. The target audience includes support technicians, developers, engineers, system administrators, and other technical users who require in-depth knowledge of the installation, configuration, and administration of a network.

1.1.1 Course goal and purpose

At the conclusion of this course, students should be able to configure their UnixWare 7 system for use on a network. By the end of the course, the student will be able to:

- Describe the various networking addresses and transmission protocols
- Install and configure a network interface card
- Configure a network for use with subnets
- Configure a system for use as a router
- Configure a WAN connection using the Point-to-Point Protocol (PPP)
- Configure a Domain Name System (DNS) server
- Configure Web and FTP servers
- Implement network security and firewalls
- Provide Windows connectivity services
- Provide electronic mail distribution

1.1.2 Target audience

This class is aimed at support technicians, developers, engineers, system and network administrators, and other technical users who operate in the UnixWare 7 environment on a regular basis. Students are likely to have a wide variety of backgrounds in UNIX®, UnixWare, and SCO OpenServer™. They must have prerequisite skills equal to or exceeding those provided in the **UnixWare 7 System Administration** courses.

Lesson 1

Course Introduction

1.2 Student Materials

The contents of the box containing your student materials are yours to keep and write notes in. Your instructor might have other reference materials in the classroom for your use while attending this class. Please do not take or write in these materials as they will be needed again for later classes.

1.2.1 The *Student Guide*

The *Student Guide* contains:

- Outline of presentation
- Note-taking space
- Copies of all transparencies

1.2.2 SCO documentation

This course refers to the following SCO printed documentation:

UnixWare 7 Getting Started Guide

Provides the information you need to get your UnixWare 7 system up and running and the *Runtime Release Notes*. These *notes* contain important late-breaking information about installations, hardware requirements, and known limitations.

UnixWare 7 System Handbook

Covers many commonly used system administration tasks, such as managing users, starting and stopping the system, working with filesystems, and configuring printers.

This course also refers to the online documentation using **scohelp(X1)**.

1.3 Typographical Conventions

This publication presents commands, filenames, keystrokes, and other special elements as shown here:

lp or lp(1)	Commands, device drivers, programs, and utilities (names, icons, or windows); the number in parentheses indicates the reference manual section in which the command, driver, program, or utility is documented
<i>/new/client.list</i>	Files, directories, desktops (names, icons, or windows), system, network, and printer names
root	User names
<i>arg</i> or <i>arg</i>	Placeholders (replace with appropriate name or value)
[<i>arg</i>]	Surrounds an <i>option</i> or <i>cmdarg</i> that is not mandatory
<i>arg ...</i>	Indicates multiple occurrences of the <i>option</i> or <i>cmdarg</i>
\$	Command-line prompt for commands executable from the Bourne shell by any user
#	Command-line prompt for commands only executable by the root user from the root account
	Keyboard keys
 	Press  and  keys simultaneously
  	Press all three keys,  ,  , and  , simultaneously
 d	Press  key. With  still depressed, press  .
 0240	Press  key. With  still depressed, press and release in sequence  ,  ,  , and  .
 Q + i	Press  key. With  still depressed, press  . Release both keys and press  .
 Q + I	Press  key. With  still depressed, press  . Release both keys and press   .
Exit program?	System output (prompts, messages)

tar tv	User input, such as on a command line
“Description”	Field names or column headings (on screen or in database)
Cancel	Button names
Edit	Menu names
Copy	Menu items
File ⇨ Find ⇨ Text	Sequences of menus and menu items
open or open(2)	Library routines, system calls, kernel functions, C keywords; the number in parentheses indicates the reference manual section in which the file is documented
\$HOME	Environment or shell variables
SIGHUP	Named constants, signals, and kernel parameters
“adm3a”	Data values
<i>employees</i>	Database names and tables
buf	C program structures
b_b.errno	Structure members

1.4 Sample exam items

These questions are representative exam items on the SCO ACE certification exam for this course. Being able to correctly answer these questions does not guarantee that you will be able to pass the actual exam.

1. A host with the IP address 186.25.22.11 has the netmask value 255.255.240.0. Which broadcast address should it use?

Note the following binary to decimal equivalents:

00001011=11
00001111=15
00010110=22
00011111=31
11110000=240

- A. 186.25.15.255
 - B. 186.25.31.255
 - C. 186.25.240.255
 - D. 186.25.255.255
2. Which file does the **routed**(1Mtcp) daemon directly read to initialize static routes to distant networks?
 - A. */etc/inet/gateways*
 - B. */etc/inet/networks*
 - C. */etc/route*
 - D. */etc/tcp*

3. Which statement regarding virtual domains is **FALSE**?
- A. Virtual domains are created using **ifconfig**(1M).
 - B. Each physical network interface can be aliased to only one virtual IP address.
 - C. The system's network services work independently, and can be configured to behave differently depending on the network interface being used.
 - D. The administrator can configure web services on the PPP interface to behave differently from web services on the Ethernet interface.

4. The following text is a portion of name server configuration file, */etc/inet/named.conf*:

```
zone "somedom.com" in {  
    type master;  
    file "db.abc.org";  
};  
  
zone "75.43.197.in-addr.arpa" in {  
    type master;  
    file "db.75.43.197";
```

```
};
```

Which record, describing host *host1.abc.org*, does **NOT** belong in the file */etc/inet/named.d/db.abc.org*?

- A. host1 IN A 197.43.75.1
- B. host1 IN MX 10 mailgate.abc.org.
- C. 1 IN PTR host1.abc.org.
- D. ftp.abc.org IN CNAME host1

Answers to sample exam items

- 1. B
- 2. A
- 3. B
- 4. C

2

TCP/IP Concepts

AV 2-1

The slide is titled "TCP/IP Concepts" in a blue-bordered box at the top. Below the title is a bulleted list of eight topics. At the bottom, there is a blue footer bar containing the text "GNAD0201" on the left and "© The Santa Cruz Operation, Inc." on the right.

TCP/IP Concepts

- Ethernet hardware addresses
- Software addresses
- Host names and IP addresses
- Routers, bridges, and gateways overview
- Translating IP addresses to hardware addresses
- Protocols and ports
- The TCP/IP protocol stack
- Other LAN hardware addresses

GNAD0201 © The Santa Cruz Operation, Inc.

Lesson 2

TCP/IP Concepts

Introduction

The UnixWare 7 environment provides a means of connecting computers from heterogeneous software environments (e.g., UNIX[®], Novell Netware, Windows, and DOS) for the purpose of exchanging data, performing remote login sessions, and mounting remote filesystems. This connectivity involves a carefully orchestrated set of system services to ensure that the entire network communicates effectively.

Understanding how these services work together is a necessary part of an administrator's job. This lesson looks at these concepts.

Objectives

This lesson describes the basic components of a computer network. By the end of this lesson, you will be able to:

- Describe hardware addressing
- Describe and use IP addressing
- Explain and use netmasks and broadcast addresses
- Explain the use of the ARP protocol
- Describe the TCP and UDP protocols and addresses
- Define the terms *client* and *server*
- Explain the function of the */etc/inet/services* and */etc/hosts* files
- List the types of network adapters supported UnixWare 7

2.1 Hardware Addresses

Computers on a network communicate with each other by sending messages between one another.

For these systems to send and receive these messages, the systems must:

- Be physically connected
- Use one or more lengths of cable to connect the systems on a *local area network* (LAN)
- Use telephone, microwave, or satellite lines to connect LANs to a *wide area network* (WAN)

Each node on a LAN has a unique hardware address. These addresses are:

- Used to effectively communicate with another system on a given network
- Referred to as the *Media Access Control* (MAC) address

- Dependent on the type of *network adapter* (sometimes called a *network interface card* or NIC) for their format. For example:

- *Ethernet* addresses:

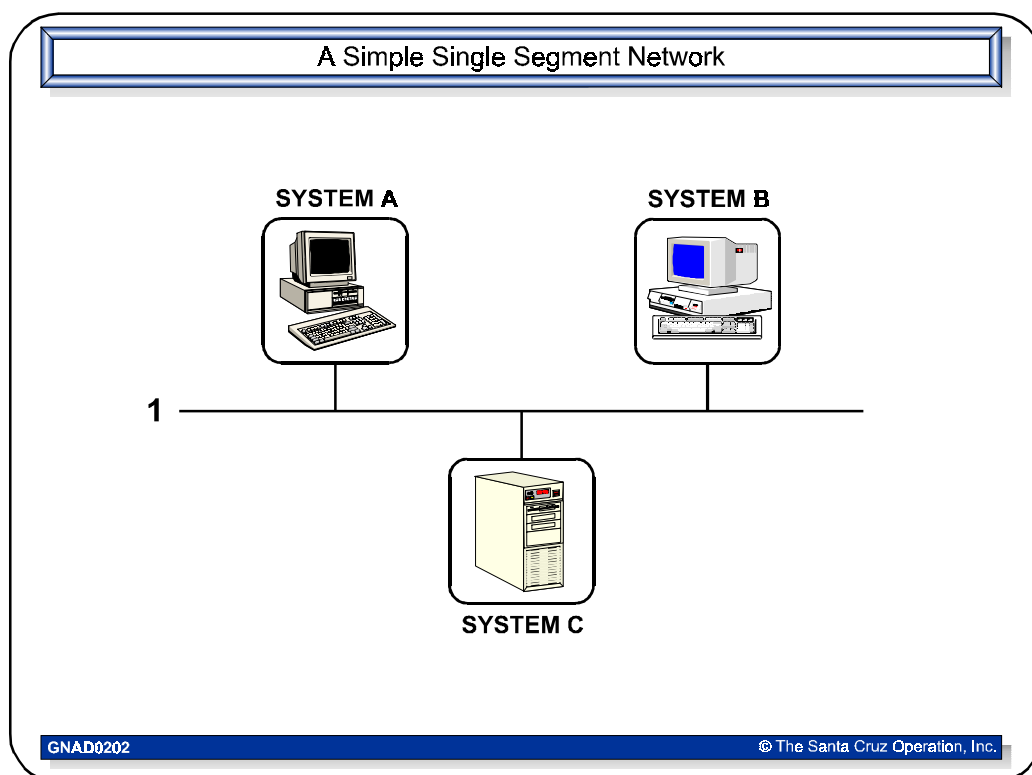
- Use a unique 48-bit (6-byte) value supplied by the manufacturer of the adapter. No two network adapters have the same Ethernet address.
- Are written as six fields of hexadecimal numbers, commonly separated by colons. For example:

00:80:5f:20:e7:d0

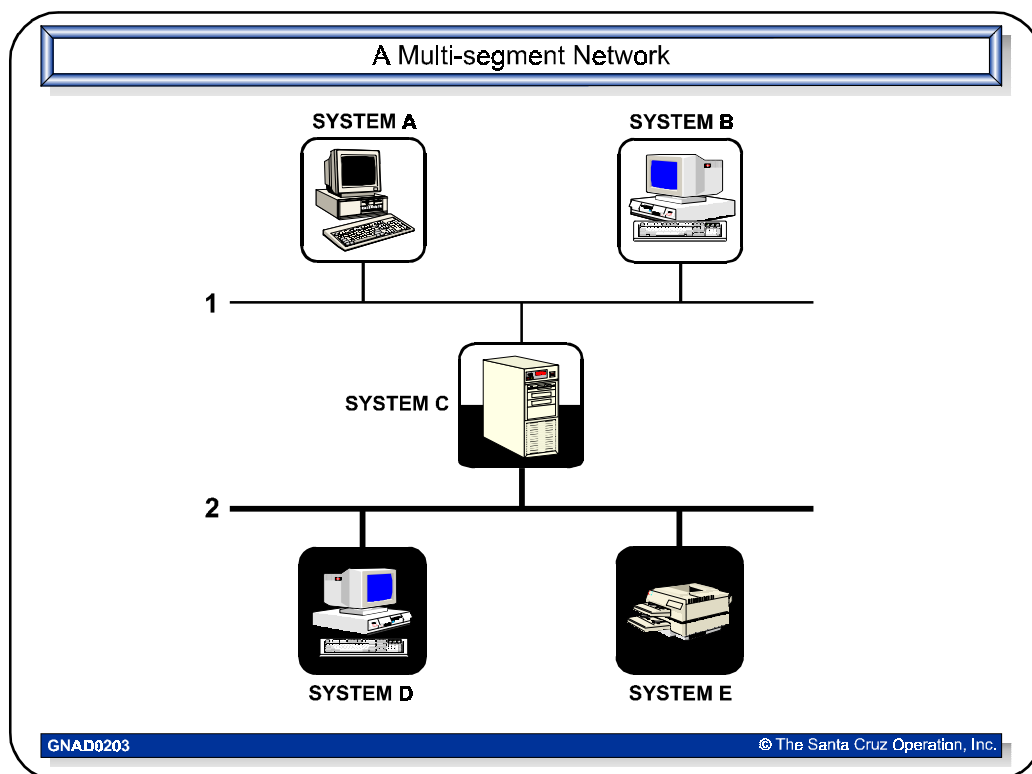
- *Token Ring* addresses are also 48-bit values, written as six fields of hexadecimal numbers. No two adapters on a given ring can have the same address. For example:

00:06:3c:31:d6:e1

AV 2-2



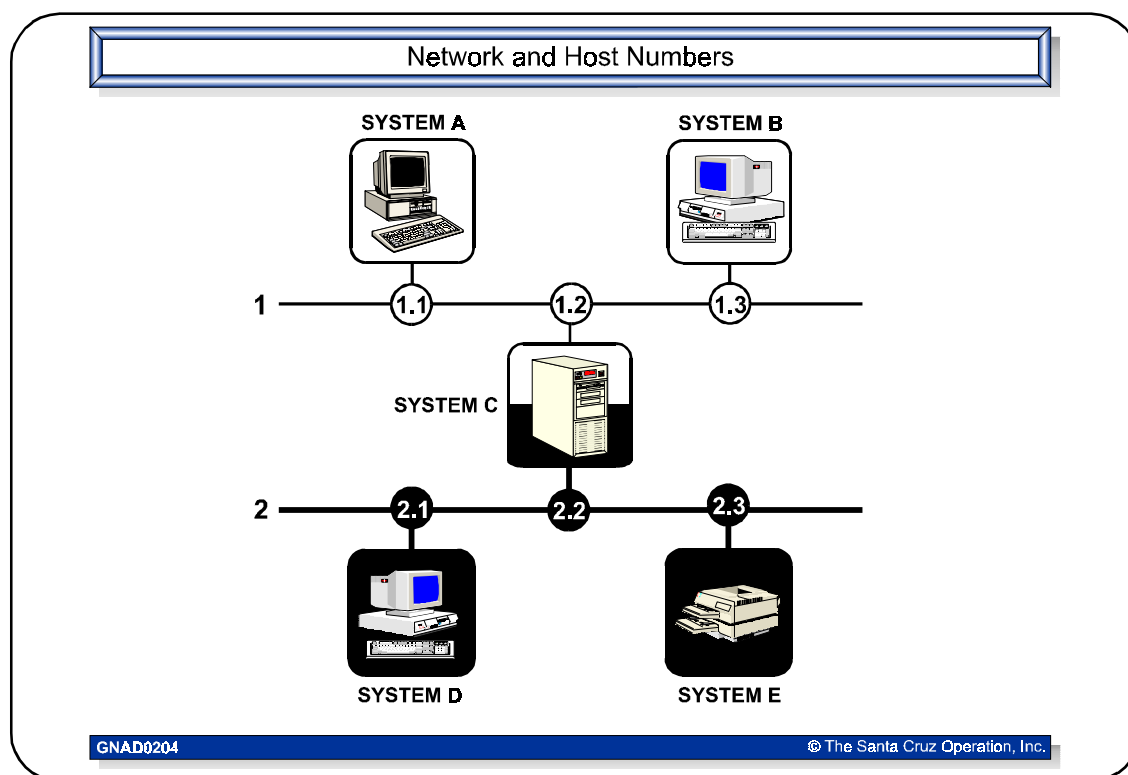
AV 2-3



Each message sent over the network uses two unique addresses:

- A *destination address* identifies which system receives the message.
- A *source address* identifies which system sent the message.

AV 2-4



2.2 Software Addresses

Software addresses can solve problems raised when using hardware addresses:

- How does a system send a message to a system on a different network?
- How are systems able to easily re-establish communications with another system that has a new network adapter and therefore, a new hardware address?

Software addresses solve these problems because they can span different networks, whereas hardware addresses are bound to a physical cable.

2.2.1 IP addresses

A unique number identifies each simple network, and each host on that network is assigned a unique number *on that network*. Therefore, every node on a network is uniquely identified by an Internet Protocol (IP) address that has the following components:

- Network number
- Host (also called *interface*) number on that network

2.2.1.1 IP address syntax

IP addresses are made up of four 8-bit numbers, called *octets*. These are written as the decimal value in each octet, separated by a period (or dot). This is called *dotted-decimal notation*. For example:

132 . 147 . 210 . 9

- Each octet is an 8-bit value and is limited to a decimal range of 0–255.
- One or more octets comprise the network portion of the IP address; one or more octets comprise the host portion. There are four octets in total.

Exercise: Counting nodes on the network

Given that each node on a network has a 4-octet (32-bit) IP address, what is the theoretical maximum number of nodes that could exist on the entire network? How did you calculate this number?

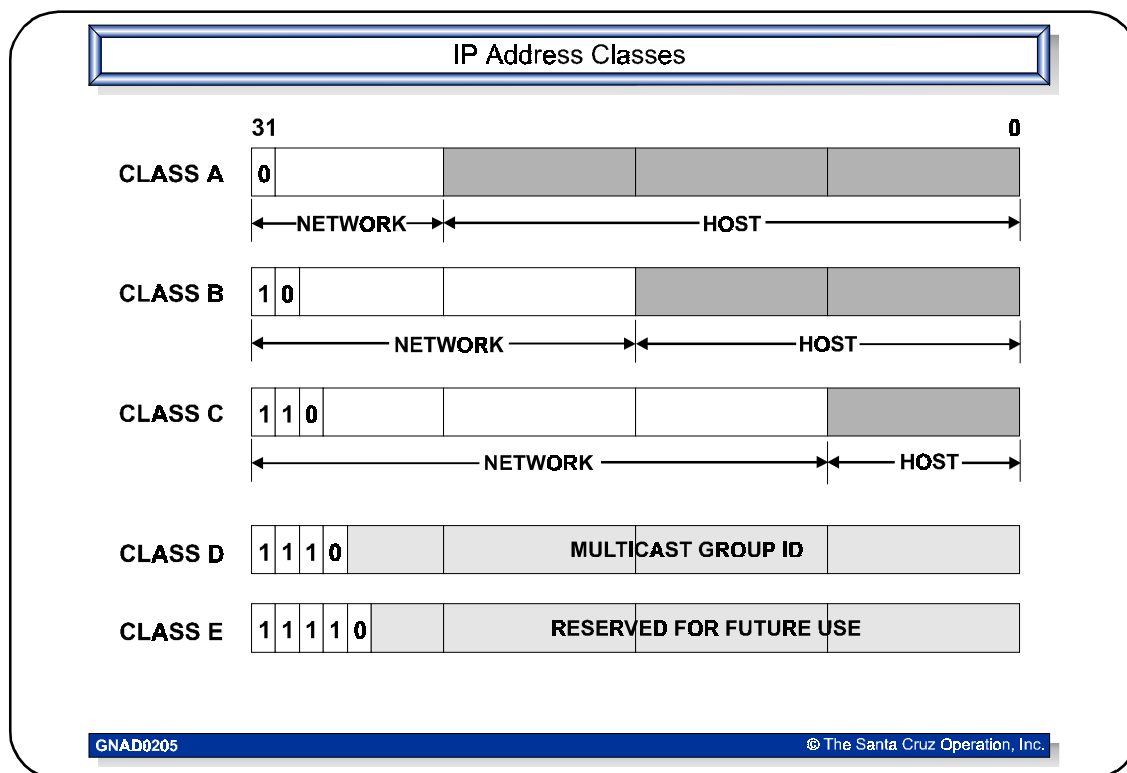
2.2.1.2 IP address classes

Recall that one part of the 32-bit IP address designates the network and the other part designates the host. The following shows the trade-offs of allocating these bits:

- If too many bits are reserved for the network portion, then you can have many different networks, but relatively few hosts on each network.
- If too many bits are reserved for the host portion, then you can have many different nodes on a given network, but relatively few networks.

To satisfy the needs of all potential network installations, the original designers of the IP addressing scheme structured these addresses into five different classes: A, B, C, D, and E. The first three classes—A, B, and C—are the most widely used.

AV 2-5



The network portion of the IP address—more specifically, the most significant (first) octet of it—specifies the class of an address. AV 2-5 shows the makeup of the different classes.

This list indicates the value for the first octet in each class:

- Class A 0–127
- Class B 128–191
- Class C 192–223
- Class D 224–239
- Class E 240–247

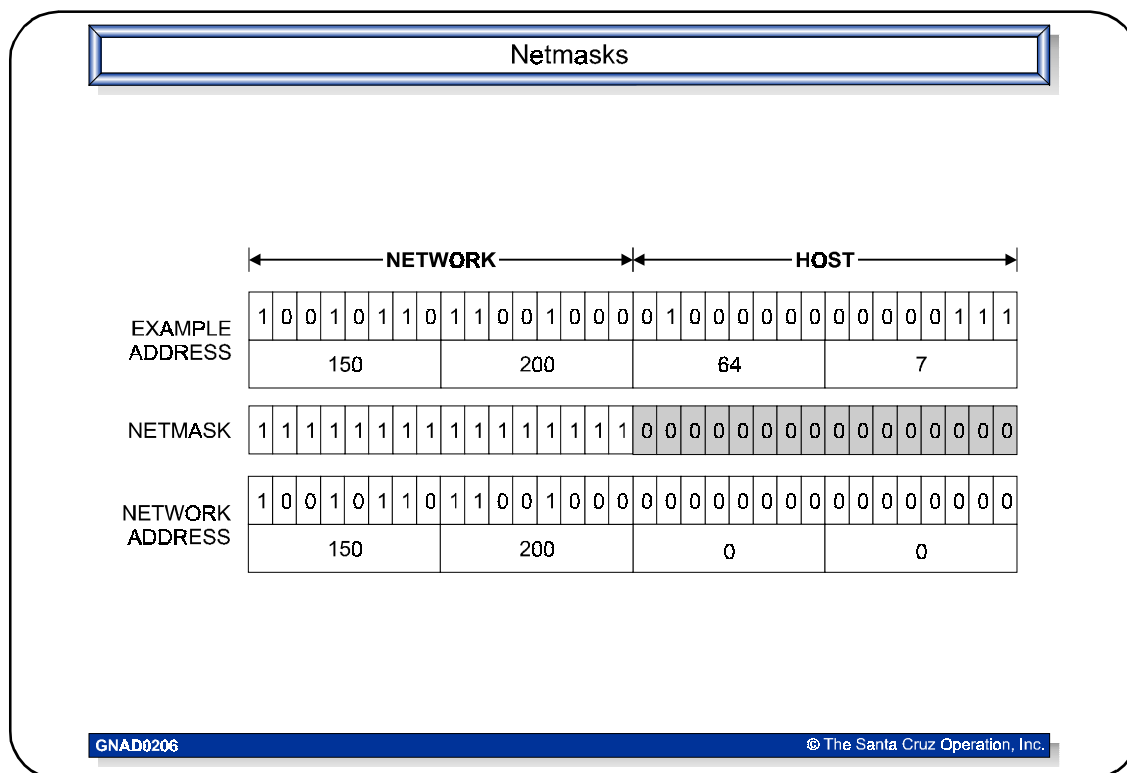
There are three types of IP addresses:

Unicast Destined for a single host

Broadcast Destined for all hosts on a given network

Multicast Destined for a set of hosts that belong to a multicast group

AV 2-6



2.2.2 Netmasks

A *netmask* indicates how many bits are used for the network address and how many for the host address. It determines the network address.

In the mask, a one (1) indicates a network address bit, whereas a zero (0) indicates a host address bit. The network address is determined by performing a logical AND of the netmask and the IP address bits. This is illustrated in AV 2-6.

The netmask can be represented with the dotted decimal notation, which is easier to use and less error prone than using a binary or hexadecimal notation.

A Class B network (that is not subnetted) uses 16 bits for the network address and 16 bits for the host address. The netmask for this network is 255.255.0.0.

Exercise: Specifying netmasks

Using dotted decimal notation, indicate the default netmask value for the following networks:

- Class A network
- Class C network

2.2.3 Network addresses

The *network address* is used to separate networks into logical groups. By default, 8, 16, and 24 bits are used for the network portion of class A, B, and C networks, respectively. Among other things, network addresses are used in routing tables (see Section 5.2.2, “Routing table,” on page 5-13).

The following table contains examples of Class A, B, and C IP addresses, netmasks, and the resulting network addresses.

Table 1: Network Addresses

Class	IP Address	Netmask	Network Address
A	110.1.2.3	255.0.0.0	110.0.0.0
B	132.147.1.2	255.255.0.0	132.147.0.0
C	200.100.50.1	255.255.255.0	200.100.50.0

These network classes can be further subdivided into *subnets*. This is accomplished by allocating some of the hosts bits to be used as address bits.

Netmasks can indicate that any number of the bits in the IP address are used for the network address.

Subnets are commonly used to simplify network administration and enable expansion. They are discussed in more detail in Lesson 4: Subnets.

2.2.4 Broadcast addresses

A *broadcast address* is used by hosts to transmit data to all hosts on a network. A *broadcast packet* is a message with a broadcast address as its destination. Broadcast packets are:

- Sent by programs that want multiple programs on other machines to pick up their data.
- Connectionless-type packets. That is, there is no acknowledgement of receipt of the packet.
- Addressed to all hosts on a network. Therefore, the packets are not normally forwarded by a router to other networks.

If you need to have a message received by multiple hosts across multiple networks, use multicasting (see Section 2.2.6, “Multicast addresses,” on page 2-21).

Examples of programs using broadcast packets include:

Routing daemons	Announces routes to other machines running routing programs.
rwhod (1Mtcp)	Announces usage metrics on the sending machine to other machines running rwhod (1Mtcp). These statistics can be seen with the rwho (1tcp) and ruptime (1tcp) commands.
ping (1tcp)	Sends packets to determine which machines are available on a network when a broadcast address is specified.

Broadcast addresses:

- Are configured for each network adapter on a host
- Consist of two parts:
 - The network portion, filled out with the number of the interface
 - The host portion, with all bits set to one (1)

For example, if a network interface has:

- IP address 132.147.20.17
- Netmask of 255.255.255.0

The broadcast address of the interface would be 132.147.20.255.

Exercise: Determining broadcast addresses

Assuming no subnetting, given the following IP addresses:

- 85.110.43.7
- 210.20.210.20
- 150.75.37.5

Determine:

- Netmask
- Network address
- Broadcast address

2.2.5 Available host addresses

The number of hosts that can exist on a network depends upon the class of the address and its netmask. Calculate this number to plan the layout of your network using this formula:

$$2^{(32-\text{the number of network bits})} - 2$$

For example, by default:

- A class A address can have $2^{24} - 2$ (16777214) hosts
- A class B address can have $2^{16} - 2$ (65534) hosts
- A class C address can have $2^8 - 2$ (254) hosts

There are two fewer hosts (2 is subtracted in the previous formula) because:

- The network address (see Section 2.2.3, “Network addresses,” on page 2-16) cannot be assigned to a host. (For example, the host portion of an IP address cannot contain all zeros).
- One of the host addresses (normally all ones) must be used as the network’s broadcast address. Broadcast addresses are discussed in Section 2.2.4, “Broadcast addresses,” on page 2-17.

2.2.6 Multicast addresses

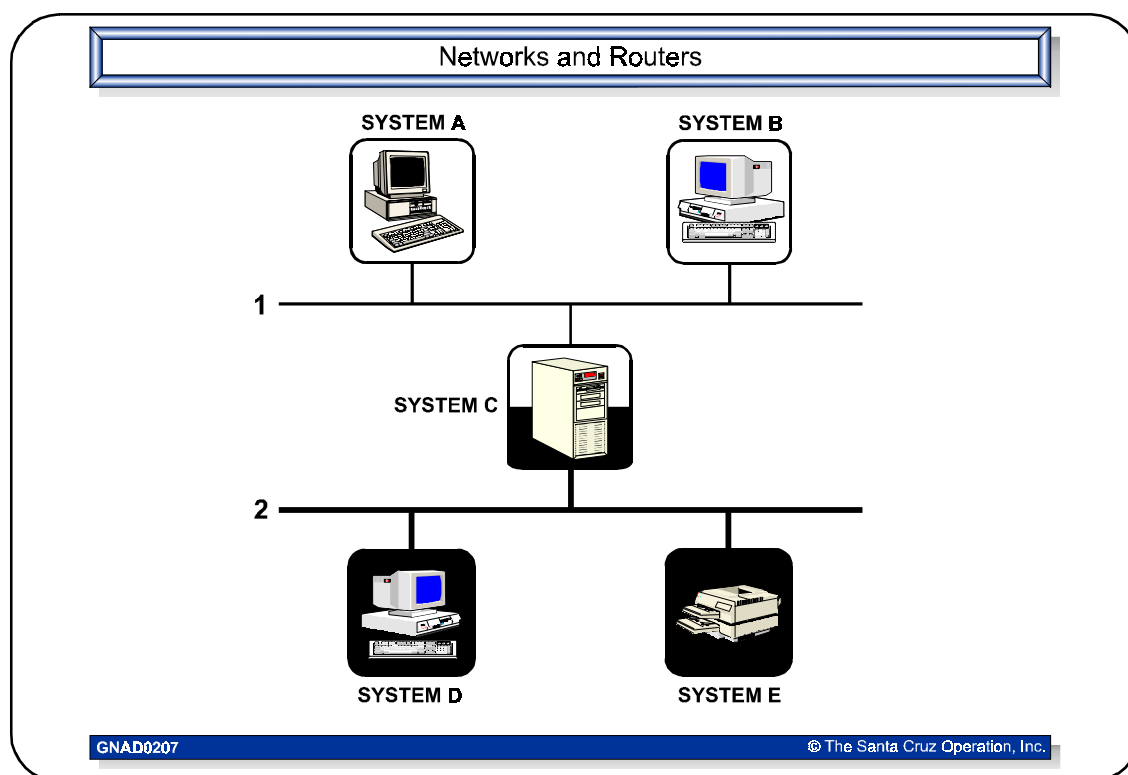
Remember that broadcast packets are normally not re-broadcast by routers. This means that you cannot use broadcast addresses to send messages to machines on multiple networks.

Multicast addresses allow applications to send data to a select group of hosts on multiple networks. An example of a multicast application is a conference where morning keynote speeches are multicast across the Internet. Audio applications are used on machines throughout the world to pick up the multicast packets and play them on their sound card.

Multicast addresses:

- Are in the D address class, which ranges from 224.0.0.0 to 239.255.255.255
- Have the address 224.0.0.0 reserved
- Have the address 224.0.0.1 assigned to the *all hosts group*, that includes all hosts and routers participating in multicasting, not just listening on a particular multicast address

AV 2-7



2.3 Network-Joining Devices

Individual network segments are often connected to create larger networks or internetworks. The alternate approach of connecting everything together on one very large network segment is not always feasible due to:

- Physical limitations on cable length
- Performance limitations for networking many nodes on one segment
- Logical limitations on the number of nodes permitted on a single segment

Options to extend your network include:

- Join multiple segments, without controlling the traffic between them.

This can overcome physical cable length limitations, but does not help performance or increase the number of hosts permitted on the segment. Examples of devices in this category include:

- A *repeater* connects two segments of a physical network to effectively extend the size of a network segment. It repeats the messages received on either side of the connected segment onto the other, without any processing of the messages.
- A *hub* is a multi-port repeater. Multiple nodes are attached to the hub, often using twisted pair (*10BASE-T* or *100BASE-T*) wiring. The hub repeats each message on every port. Hubs range from as few as 4 to as many as 32 or more ports.
- Join multiple segments, controlling the traffic between them.

This can overcome physical cable limitations and help improve network performance. If a message is sent to a node on the local segment, it is not necessary to repeat it to the other segments.

A non-local message is only repeated to the segment of the destination node. This technique allows multiple segments to be used for different messages at the same time. However, it does not increase the number of nodes permitted on the segment. Examples of devices in this category include:

- A *bridge* is an intelligent repeater. It receives messages on one network, and only passes the message to the other network if necessary.
- A *switch* is an intelligent hub. It only repeats a message on the port to which the destination node is attached. A switch can handle multiple simultaneous connections between pairs of nodes.

- Join multiple logical networks

This option overcomes limitations on the number of nodes permitted on each segment, in addition to overcoming physical restrictions of cable length. It can also improve performance, because local traffic on each network is not re-sent on the other networks. Examples of devices in this category include:

- A *router* is a multi-homed host— a host with two or more network connections. Routers behave differently than single-homed hosts when they receive network messages whose IP address does not match their own. Single-homed hosts discard the message. Routers pass the message along to the next router or directly to the destination host if possible.
- A *gateway* is a specialized router that translates network messages from one protocol to another (e.g., between IP and IPX). In some contexts, the term gateway is used interchangeably with the term router.

See Lesson 5: “Routing,” for further information on how to configure a host to act as a router.

2.4 Host Names and IP Addresses

IP addresses are the basis for the routing of data between hosts in a TCP/IP-based network. However, people easily remember and use host *names* than IP *addresses*. Some software is designed to work only with host names.

Host names are mapped to IP addresses so that both people and the IP routing software can use the information:

- A host name must be mapped to an IP address, so that the IP layer of the TCP/IP protocol stack can determine to which network host to direct the data.
- The mapping of a host name to an IP address is done via the *resolver*.
- The resolver traditionally uses the */etc/hosts* file.

This section describes how to configure host name to IP address mapping.

2.4.1 System name

Your system or host name should be unique on your network and have these properties:

- Consist of lowercase letters, numbers, and dashes
- Begin with a letter
- Contain no more than 63 characters

Some older systems might be limited to seven characters. If you are using your system in a network with these types of systems, you should limit your host name to seven characters.

2.4.2 Domain names

With regard to host names, the term *domain* is used to describe a group of machines that all have the same owner, type of owner, or the same geography. For example, in the United States:

- Non-profit organizations have machines in the *org* domain.
- Commercial (for-profit) organizations have machines in the *com* domain.

Each domain is divided into a sub-domain, normally named after an organization. For example:

- All machines belonging to the United Nations are in the *un.org* sub-domain.
- All machines belonging to The Santa Cruz Operation (SCO) are in the *sco.com* sub-domain.

Users and administrators call both directories and sub-directories by the common term *directory*. Similarly, network administrators call both domains and sub-domains by the common term, *domain*.

You can assign a domain name to each machine added to your network to simplify networking tasks for both users and administrators. Usually, all machines on your network will have the same domain name.

A host name with its domain name appended on the end is called a *fully qualified host name*.

AV 2-8

The */etc/hosts* File

```
$ cat /etc/hosts
#      @(#)hosts      6.1 8/21/93 - STREAMware TCP/IP  source
#      SCCS IDENTIFICATION
127.0.0.1      localhost
132.147.193.67 cousin.pdev.sco.com cousin
132.147.65.216 cheydev.pdev.sco.com cheydev
132.147.210.9 websco.sco.com websco www.sco.com
```

GNAD0208

© The Santa Cruz Operation, Inc.

2.4.3 The */etc/hosts* file

The */etc/hosts* file (see *hosts(4tcp)*) contains a list of hosts that share the network, including the local host. It is referenced by programs that need to translate between host names and IP addresses when a name server or *Network Information Service* (NIS) is not being used.

Each line in the file describes a single host on the network and consists of two or more fields separated by any number of blanks or tabs:

IP_address name [aliasname ...]

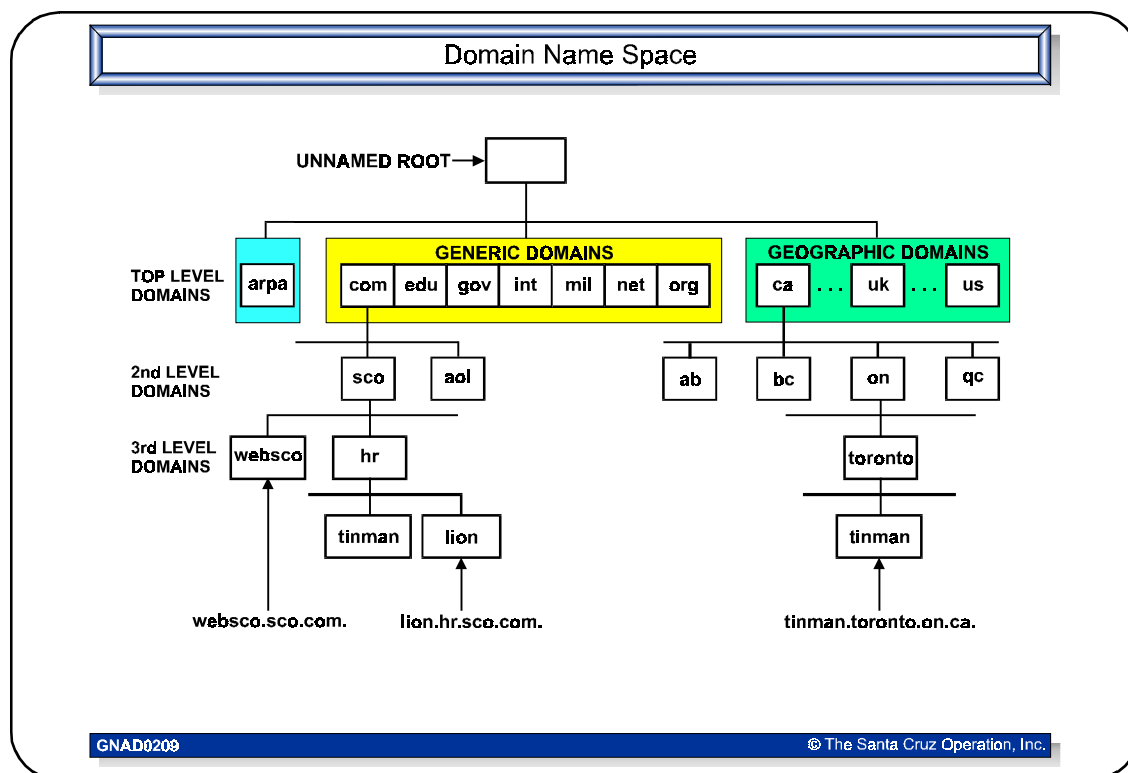
- The *IP_address* must be the first item on a line; there cannot be any characters (e.g., spaces or tabs) preceding it.
- The *name* is a fully-qualified domain name.
- The *aliasname* is not required in most cases.
- The routines that use this file ignore comments (portions of lines beginning with a pound (#) character) and blank lines.

NOTE: *localhost* (IP address 127.0.0.1) is a special name *on every machine*. It is used for testing the TCP/IP software and for communication to the same system (using the *loopback* interface) through TCP/IP. Do not modify this line in the */etc/hosts* file.

Exercise: Viewing */etc/hosts*

View the */etc/hosts* file on your system. If your network card has not been configured, the only entry will be *localhost*.

AV 2-9



If your network has many nodes, it can be very cumbersome to update the `/etc/hosts` file on each system whenever changes are made to the network. In addition to (or in place of) the `/etc/hosts` file, you can solve this problem by using:

- The Domain Name System (DNS) to resolve host names into IP addresses. DNS is discussed in Lesson 8: “Domain Name System (DNS) Configuration.”
- The Network Information Service (NIS) to resolve host names into IP addresses. NIS configuration is beyond the scope of this course.

The resolver can be configured to use all of these resources for host-to-IP resolution.

2.4.3.1 Domain name space

Section 2.4.2, “Domain names,” on page 2-27 discussed the concept of a domain. Under the Domain Name System (DNS), the Internet consists of a hierarchy of domains called the *domain name space*. It is organized as an inverted tree, similar to the UNIX filesystem.

- The *root* domain—the root of the tree—is the top-level domain.
- The top level of the domain name space is well defined. You configure branches of the tree at individual sites after you register them. (See Section 8.2, “Configuring DNS Name Servers,” on page 8-17.)

The top-level domains are divided into three areas:

- *arpa* is a special “domain” used for address-to-name mappings.
- Seven 3-character generic domains (sometimes called organizational domains):
 - Commercial organizations (*com*)
 - Educational institutions (*edu*)
 - International organizations (*int*)
 - U.S. military (*mil*)
 - U.S. governmental organizations, excluding the military (*gov*)
 - Networks (*net*)
 - Other non-profit organizations (*org*)

- Two-character geographic domain names, based on the country codes found in ISO 3166. Examples include:
 - Europe, e.g., the United Kingdom (*uk*), Germany (*de*), France (*fr*)
 - North America, e.g., Canada (*ca*), the United States (*us*)

Many countries form second-level domains beneath their country code similar to the generic domains (for example, *.ac.uk* is for academic institutions in the United Kingdom; *.co.uk* is for commercial organizations in the United Kingdom).

Refer to **scohelp** ⇨ **Internet and Intranet** ⇨ **Configuring Domain Name System (DNS) Servers** ⇨ **The Internet domain structure**.

2.4.3.2 Using domain names

Although the hierarchical structure of the domain name space is similar to the hierarchical structure of the UNIX filesystem, paths to nodes at the bottom of the “tree” are specified differently:

- For a UNIX filesystem, you start at the root directory of the tree and go down, separating each level with a slash (/) character. For example:

/etc/conf/cf.d/stune

- For the domain name space, you start at the node and go *up*, separating each level with a dot (.) character. For example:

lion.hr.sco.com

A fully qualified domain name (also called an *absolute domain name*) includes the complete domain name as well as the host name. This fully qualified name also ends with a dot. For example:

lion.hr.sco.com.

2.4.3.3 Setting the domain name

The domain name is set during the initial system load (ISL) and is recorded in several places, such as:

- Mail configuration files
- DNS configuration files
- NIS configuration files
- Web server configuration files
- The */etc/hosts* file

Changing the domain name of a machine involves modifying all locations individually. Therefore, it is important to set the domain name properly at Initial System Load (ISL).

2.5 Translating IP Addresses to Hardware Addresses

Recall that users specify a host name when connecting to another host. This name is then mapped to an IP address using the */etc/hosts* file (or other network service). The IP address must then be mapped to the hardware address of the remote system; this is done by an address resolution protocol.

2.5.1 Address resolution protocol (ARP)

The *address resolution protocol* (ARP) translates IP addresses to hardware addresses. ARP works as follows:

1. The requesting host broadcasts a message to all hosts on the local network asking the owner of the IP address to respond with their hardware address.
2. The host that has the IP address responds with its hardware address using a unicast message to the asking machine.

2.5.2 The ARP cache and the **arp(1Mtcp)** command

Each machine in an ARP request caches the other's hardware address in kernel memory. This area of memory is referred to as the *ARP cache*.

This cache is able to increase overall efficiency because:

- Machines do not need to repeat broadcasts for information about systems with which they have recently communicated. This reduces both network traffic and latency in initial message exchanges.
- The ARP request includes the IP-address/hardware-address pair of the requesting system, which the receiving system stores in its own ARP cache. Therefore, the receiving system does not have to broadcast an ARP request itself to get this information if it needs to initiate a conversation with the other system.

The arp cache can be viewed with the **arp(1Mtcp)** command. This command:

- Displays and modifies the Internet-to-Ethernet address translation table, normally maintained by the address resolution protocol (ARP)
- Has a general syntax of:

```
arp [-option] hostname
```

- Can be used to display the entry for a individual host if only the host name is given. For example, the following command displays the current ARP entry for *mother*:

```
$ arp mother
mother (132.147.65.18) at 8:0:9:69:ac:46
```

- Displays all entries in the ARP table with the **-a** option. For example:

```
$ arp -a
mother.dev.sco.COM (132.147.65.18) at 8:0:9:69:ac:46
father.dev.sco.COM (132.147.65.19) at 0:80:5f:20:e7:d0
teacher.dev.sco.COM (132.147.65.190) at 0:0:c0:92:4b:24
```

Refer to **arp(1Mtcp)** for other options.

2.6 Protocols and Ports

On a typical system, several network-oriented applications or processes can be running concurrently. Every process on any given machine uses the same protocol stack to communicate with other processes on the network.

This section describes how the protocol stack sends data to the correct process.

2.6.1 Clients and servers

An application that offers a service that can be reached over a network is called a *server*. It accepts requests over the network, performs its service, and then returns the result to the requester. Some simple services can require only a single IP *datagram* (a network transmission unit) for a request and another one for the reply; other services can require multiple datagrams to be exchanged.

A process becomes a *client* when it sends a request to a server and waits for a response. This *client-server* model parallels interprocess communications (IPC) that occurs on a single machine. Network applications are developed using the IPC model.

2.6.2 Ports

When a system communicates with another, a process on the local system communicates with another process on the remote system. This introduces two key questions regarding this exchange:

- How does the local process (the client) know which process to talk to on the remote system (the server)?
- How does the remote process (the server) know who sent the message (the client) so it can reply if necessary?

The mechanism for exchange is called a *port*. A given port number refers to a particular process on a networked computer. Some services are provided at pre-determined network ports. These “well-known” services are defined in the */etc/inet/services* file.

Refer to *services(4tcp)* for more information on */etc/inet/services*.

Port values less than 1024 are special because only the **root** user can listen in on these ports, providing for more secure services. New services added to the system must recognize this feature:

- If the new port value is less than 1024, the server process must run as **root**, and therefore has the potential to exploit the system (especially if the server software is poorly designed).
- If the new port value is greater than or equal to 1024, the server software can run as any user.

2.6.3 Transport protocols

There are two primary transport protocols used by TCP/IP applications:

- The *Transmission Control Protocol* (TCP) uses the combined specifier of internetwork address and port number to uniquely identify internetwork addresses. This combination is called a *socket* or an *end point*.

TCP offers connection-oriented, reliable stream service between two hosts on the network. It is the primary transport protocol used by TCP/IP applications.

- The *User Datagram Protocol* (UDP) is a simpler, connectionless transport-layer protocol used to exchange requests and replies between two networked hosts. Unlike TCP, it offers no reliability guarantees.

Refer to **tcp**(7tcp) and **udp**(7tcp) for more information on these protocols.

Exercise: Identifying ports and protocols

Recall that several well-known services are defined in */etc/inet/services*. Examine this file and indicate the port number of the following services. Do these services use the TCP or UDP protocols?

- **netstat**
- **ftp**
- **who**
- **login**

2.7 The TCP/IP Protocol Stack

Transmitting data across networks is a very complicated task. This complex task is usually broken down into simple components that build on one another to accomplish the entire task.

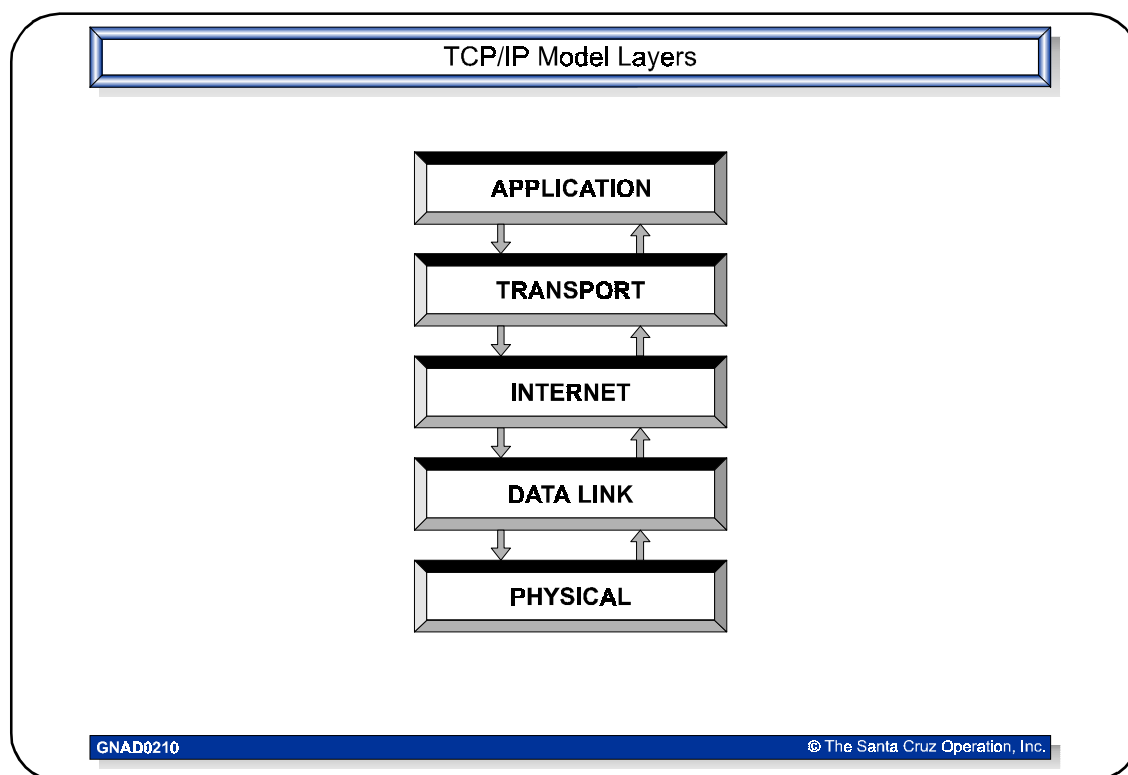
To manage the diversity of tasks required by networks, most networking software divides the requirements into a series of layers.

The rules and conventions implemented at each layer are known collectively as the layer's *protocol*.

Each protocol layer offers certain services to the layer directly above or below it.

The number, name, contents, and function of each layer differ between network models.

AV 2-10

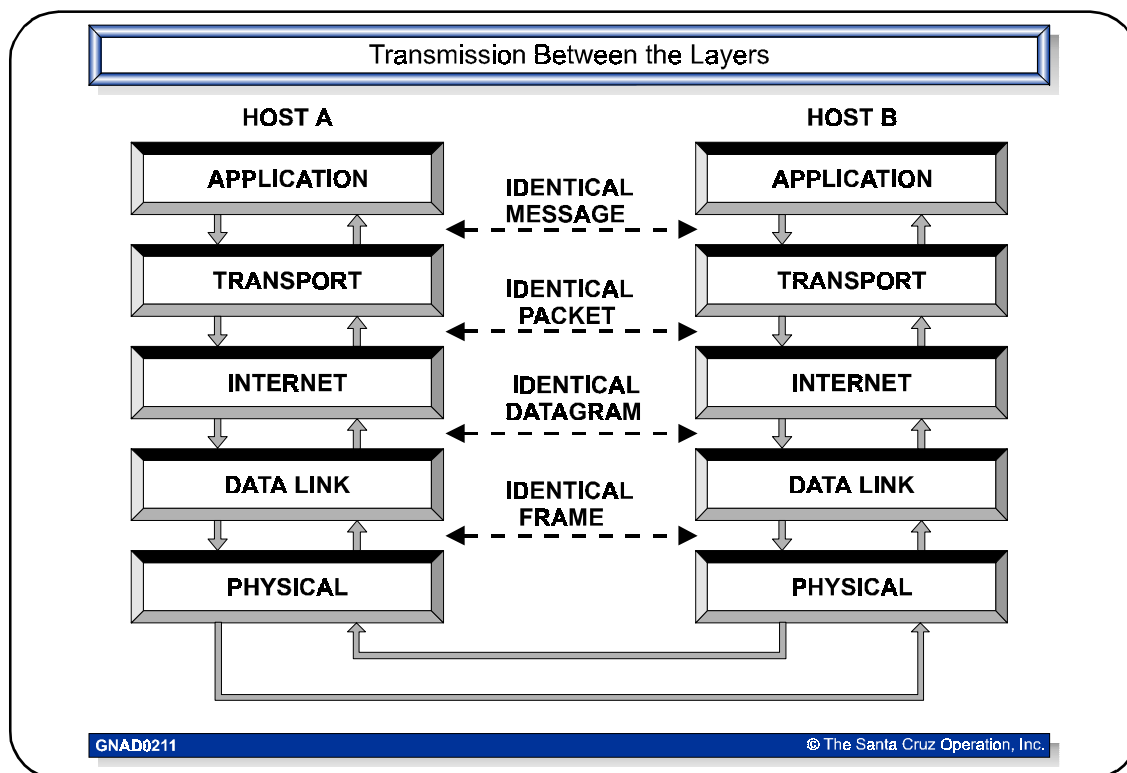


2.7.1 TCP/IP layers

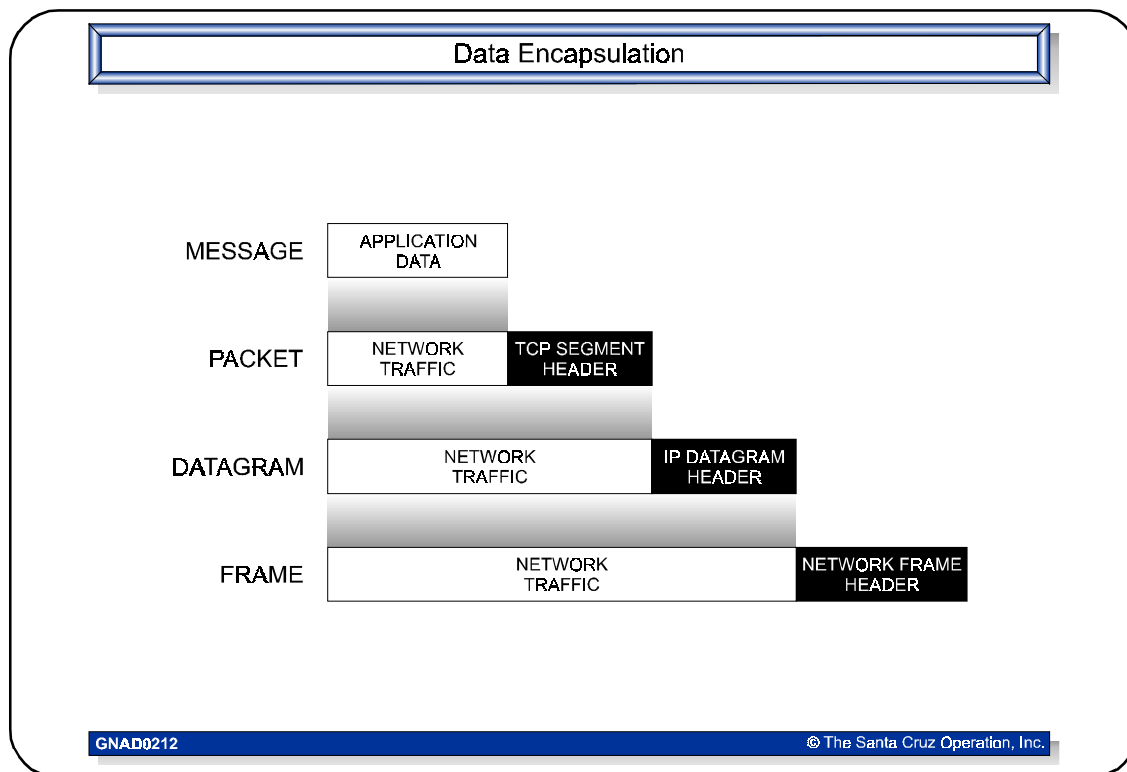
The TCP/IP model uses five layers in its protocol stack. These are shown in order in AV 2-10:

<i>Application layer</i>	The user interacts with the network application. Data comes in as commands from the user and also from network applications on the other end of the connection. TCP/IP applications work in client/server pairs at this layer.
<i>Transport layer</i>	Manages the flow of data between the hosts. The TCP and UDP protocols are implemented at this layer.
<i>Internet layer</i>	Data is moved around the internetwork. The Internet Protocol (IP) operates at this layer to route packets across the network. This layer is sometimes called the <i>network</i> layer.
<i>Data link layer</i>	Data is transmitted across a single, specific network. This layer consists of a device driver and the network adapter. This layer is sometimes called the <i>network interface</i> layer. The drivers supported in UnixWare 7 are discussed in Lesson 3: “Configuring a Network Adapter.”
<i>Physical Layer</i>	This layer consists of the physical hardware connection between hosts.

AV 2-11



AV 2-12



2.7.2 Transmission between the layers

As data moves down the protocol stack from the application layer to the physical layer, each intermediate layer adds information to the original data. These headers are encapsulated such that each succeeding protocol layer wraps its own header around the data that will be delivered.

On the receiving system, the process is reversed. Each protocol layer strips off its own header, processes the data for that layer, and then passes the result up to the next layer.

The TCP/IP protocols were designed with the *layering principle*, which states that Layer *n* at the destination receives exactly the same object sent by Layer *n* at the source. This is shown in AV 2-11.

The objects sent at the different layers have different names:

- *Messages* are passed between the application and transport layers.
- *Packets* are passed between the transport and Internet layers.
- *IP datagrams* are passed between the Internet and data link layers.
- Network-specific *frames* are passed between the data link and physical layers.

The objects are passed from layer to layer using a process of *encapsulation* that encloses one network protocol into another network protocol by adding specialized header information.

2.7.3 An example transmission

This **rlogin**(1tcp) command line illustrates how the TCP/IP protocol stack functions:

```
rlogin remote_host_name
```

2.7.3.1 Application layer

At the application layer, the **rlogin** command is a user interface (client) to the **rlogind**(1Mtcp) daemon (server) running on a remote system. The **rlogind** daemon provides a login session service.

rlogin looks up the *remote_host_name* in */etc/hosts*, DNS, or NIS using a set of library routines called a *resolver*.

- If the *remote_host_name* is not found, an error message is generated.
- If the *remote_host_name* is found, it is translated to an IP address.

2.7.3.2 Transport layer

Recall the two protocols at the transport layer of the TCP/IP protocol stack:

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

At the transport layer, the **rlogin** command uses the TCP protocol for performing I/O to the network. So that all the data for the **rlogin** command is delivered only to itself, **rlogin** obtains a TCP network port. To do this:

- The **rlogin** command needs to know which port the **rlogind** (server) is using for network I/O, it looks up the login service in */etc/inet/services* file (the login service is a “well-known” service).
- On the recipient system, the TCP software determines which process receives the data, based on the TCP port number.

2.7.3.3 Internet layer

At the Internet layer, the IP address is translated to a physical (hardware) address as follows:

1. It checks the ARP cache to see if an existing translation exists.
2. If no translation is found, the ARP software sends a broadcast packet to resolve the hardware address.
3. The remote system responds with its physical address that is entered into the local ARP cache.
4. The IP software on the recipient system uses the IP address to verify that the incoming datagram is destined for the host.
5. It then passes the datagram (stripped of the IP header) to the transport layer as a packet.

2.7.3.4 Data link layer

The data link layer forms the complete physical packet and transmits it on the physical media (cable).

On all systems, this layer checks the destination physical address of the frame to determine if the frame is addressed to this host. Only one system will accept the frame and pass it up to the Internet layer, stripped of the physical (Ethernet) header.

2.7.3.5 Physical layer

The physical layer consists of the network cards and media cable that connects the systems together in the network.

2.8 LAN Hardware

Section 2.7.3.5, “Physical layer,” on page 2-49 indicates that the bottom layer in the TCP/IP protocol stack is the physical layer, consisting of network adapters and associated cables (if any). So far in this lesson, almost every hardware example has used Ethernet at the physical layer.

This section discusses LAN technologies including:

- Ethernet—A wide variety of network adapters are supported in UnixWare 7.
- IBM Token Ring— A selection of network adapters are supported in UnixWare 7.
- Fiber Distributed Data Interface (FDDI)— A few network adapters are supported in UnixWare 7.
- Asynchronous Transfer Mode (ATM)—Supported only by the vendor.

Serial connections are discussed in Lesson 6: “Configuring WAN Connections.”

NOTE: If you use more than one type of networking hardware in your LAN, you must place an appropriate device between them. This device is normally a router. See Section 2.3, “Network-Joining Devices,” on page 2-23 and Lesson 5: “Routing.”

2.8.1 Ethernet

Ethernet is the most commonly-used physical layer. In its simplest form, Ethernet:

- Connects all workstations with a single coaxial wire, much like an I/O bus, on which all machines transmit and receive data.
- Controls access to the wire using a protocol known as Carrier Sense Multiple Access with Collision Detection (CSMA/CD). This allows hosts to listen and determine when to transmit packets. When collisions occur, the host waits and retransmits later.

The actual physical media used by Ethernet includes:

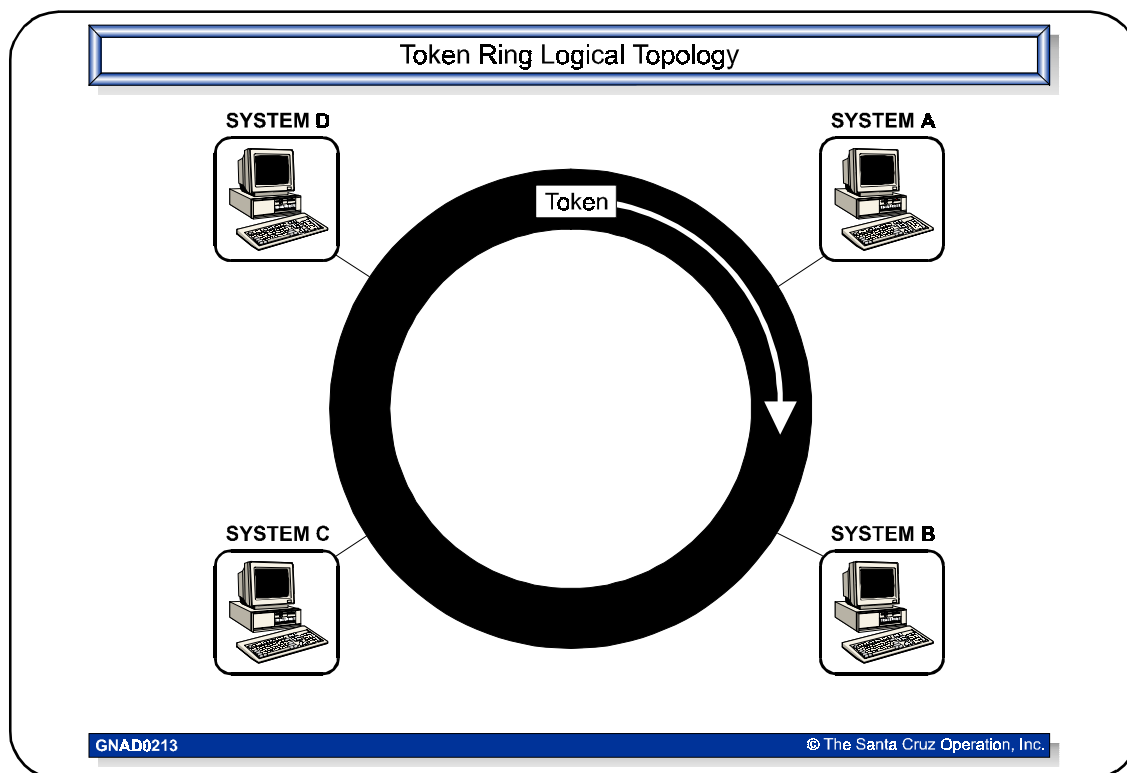
10BASE5 (Thicknet)	Uses 1/2 to 3/8ths inch coaxial cable. Hosts connecting to thicknet must also have external transceivers. Thicknet transmits at 10Mbps and is infrequently used today.
--------------------	--

Thicknet is limited to 500 meters per segment; maximum 5 segments; 2500 meters maximum cable; maximum 100 stations per segment; 300 total stations.

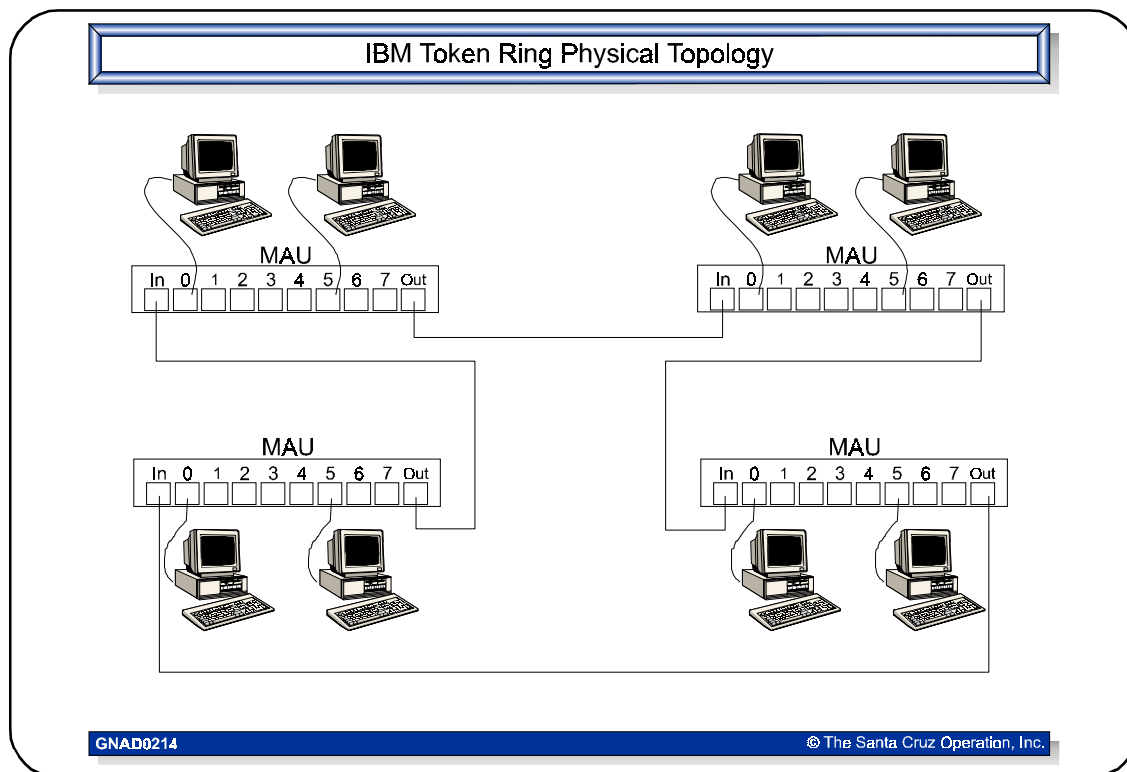
10BASE2 (Thinnet)	Uses 1/4 inch coaxial cable with BNC and “tee” connectors. Like thicknet, thinnet transmits at 10Mbps and is being phased out, usually in favor of 10BASE-T. Thinnet is limited to 185 meters per segment; maximum 5 segments; 925 meters maximum cable; maximum 30 stations per segment; 90 total stations.
10BASE-T (Twisted Pair)	Uses an 8-wire cable and RJ45 connectors. The cables are rated by category from 1 to 5. You should have at least a category 3 cable for 10BASE-T because it transmits at 10Mbps. A separate cable is run from each host using 10BASE-T to a compatible hub or switch. 10BASE-T is limited to 100 meters per segment; maximum 5 segments; 100 meters between hubs; 100 total stations.
100BASE-T	Very similar to 10BASE-T, except that it requires category 5 cables as it transmits at 100Mbps. 100BASE-TX (2 pair cable) is limited to 100 meters per segment.
Gigabit Ethernet	Transmits at 1Gbps in a switched environment.
Radio Frequency (RF)	Uses radio waves to send and receive data. RF speeds vary from 10 to 100Mbps.

A common optimization in Ethernet LANs is the use of switches because they dramatically decrease collisions. For more information on switches, see Section 2.3, “Network-Joining Devices,” on page 2-23.

AV 2-13



AV 2-14



2.8.2 IBM Token Ring

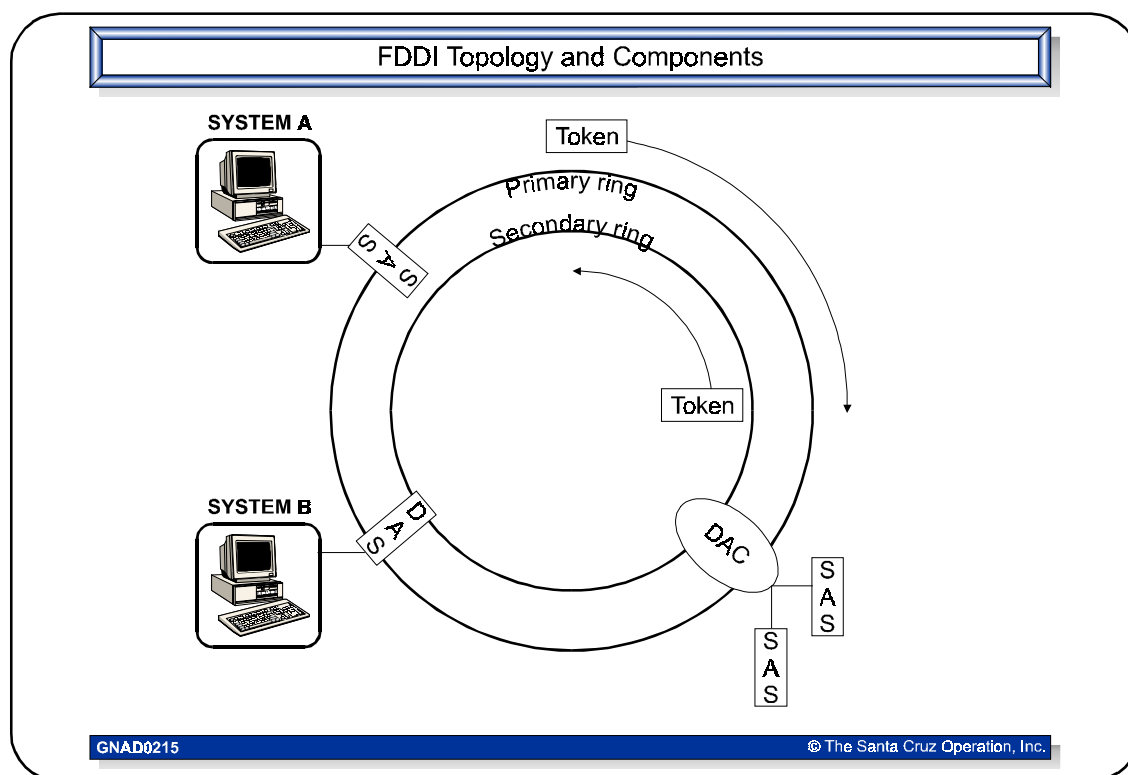
Token Ring was originally developed by IBM to overcome some of the performance problems Ethernet experiences when a segment becomes overloaded.

Compared to the Ethernet architecture, Token Ring:

- Suffers much less performance degradation with large amounts of traffic
- Is best suited for highly populated LANs

Each workstation is connected to a Multiple Access Unit (MAU), which supports up to eight workstations. Additional MAUs can be cascaded to support more workstations, as illustrated in AV 2-14.

AV 2-15



2.8.3 FDDI

Fiber Distributed Data Interface (FDDI) is another token-ring based technology. Because FDDI uses light to carry the signals in fiber-optic cables (instead of wires), FDDI is useful if you have to work with any of these limitations:

RFI	Radio Frequency Interference (RFI) is radio waves created by electrical equipment (e.g. electric motors and generators). RFI does not affect the light traveling through fiber.
Security	Because light does not radiate RF signals, FDDI can not be monitored by radio-based eavesdropping devices. This increases security.
Distance	FDDI cables can extend up to 6.2 miles (10KM.) This is much farther than any of the other cable-based physical layers.

2.8.4 ATM

For some time, Asynchronous Transfer Mode (ATM) was the fastest physical layer available. Its chief benefit is that it has no distance limit. The problems with ATM are:

Expense	ATM has not been widely received and so it has not reached reasonable economies of scale.
Administration	ATM uses virtual connections and the LAN administrator must set up these virtual circuits between all possible machines that want to communicate on a LAN.

Current market trends indicate that gigabit ethernet is being adopted for fast LAN connections. ATM, because it has no distance limit, is being used for fast WAN connections.

2.9 The Future of the Internet Protocol (IPv6)

The Internet Protocol described so far in this course is known as IPv4 (Internet Protocol Version 4). It has changed little since IP was devised in the 1970s, and now presents several problems, including:

- The 32-bit address space provides for more than 10^9 addresses. However, the addresses are not allocated for efficiency so this is too small. The InterNIC is running out of addresses.
- The protocol does not provide features necessary to meet the demands of new network-oriented applications, for example, synchronizing the sound and video of a multimedia presentation being multicast across the Internet.

Although there have been stop-gap measures to solve some problems, the ultimate solution to the shortcomings of IPv4 is the next generation of the protocol known as IPv6.

Some of the basic features of IPv6 are:

- 128-bit addresses, which provides for more than 10^{38} addresses
- Options to allow synchronization of packets that are useful for multimedia
- Reserved bandwidth for high-priority data with a priority scheme from 0-15
- Secure data interchange according the IPSEC encryption specification

UnixWare 7 contains application libraries that allow application programmers to create IPv6-aware programs. However, all IPv6 packets have source and destination addresses which are IPv4-mapped IPv6 addresses (see RFC1884). A future release will have a full IPv6 implementation.

2.10 Lesson Summary

The UnixWare 7 environment allows computer systems to use a network to exchange data, perform remote login sessions, and mount remote filesystems. This lesson described the mechanics of network addresses—both hardware and software—and how a network host uses them to identify another host on the network.

Review questions

1. What is the difference between a hardware (Ethernet) address and an IP address?
2. What are the components of an IP address?
3. What is the purpose of the netmask?
4. To which class does the address 125.110.65.3 belong? What broadcast address does it use (assuming there is no subnetting)?
5. How many different Class A, B, and C networks are available? (Hint: remember that the network portion of the IP address cannot be all zeros or all ones.)
6. What are the seven generic domains?

7. What is the purpose of a router?
8. What is the purpose of address resolution protocol (ARP)? Describe how it functions.
9. What are the five layers of the TCP/IP protocol stack?
10. What is the difference between the TCP and UDP protocols?
11. What do the terms *client* and *server* mean?
12. What is the purpose of the */etc/hosts* and */etc/inet/services* files?

2.11 Relevant Commands, Files, and Syntax

The following commands and files were presented in this lesson.

Relevant commands

arp (1Mtcp)	Displays and modifies the Internet-to-Ethernet address translation table used for the address resolution protocol (ARP)
ndstat (1M)	Displays statistics for configured network adapter drivers
rlogin (1tcp)	Is the remote login client, that connects your terminal on the current local host to a remote host system
rlogind (1Mtcp)	Is the server for the rlogin (1tcp) program
inetd (1Mtcp)	Is the Internet “super-server,” that listens on multiple ports for incoming connection requests and then spawns the appropriate server processes to handle them

Relevant files

<i>/etc/inet/inetd.conf</i>	Is the configuration file for inetd (1Mtcp). See also <i>inetd.conf</i> (4tcp).
<i>/etc/hosts</i>	Lists the host names and IP addresses of the hosts that share the network. See <i>hosts</i> (4tcp).
<i>/etc/inet/services</i>	Lists the service name, port number, protocol name, and aliases of well-known network services. See <i>services</i> (4tcp).

2.12 Answers to Review Questions

1. A hardware (Ethernet) address is a unique 48-bit (6-byte) address value determined by the network adapter manufacturer. An IP address is a 32-bit (4-byte) address value that specifies the network and host addresses for a given network node on the Internet.
2. The components of an IP address include the network portion and the host portion. (The host portion can be further subdivided into a subnet address and host address if subnetting is configured.)
3. The netmask specifies which portion of the IP address refers to the host address and which portions refer to the network (and subnet) addresses.
4. The address 125.110.65.3 is a Class A address. Its broadcast address is 125.255.255.255.
5. Because the assignable component (not the first bits which are predetermined) of the network portion of the IP address cannot be all zeros or all ones:

The number of available class A networks is $2^7 - 2 = 126$

The number of available class B networks is $2^{14} - 2 = 16,382$

The number of available class C networks is $2^{21} - 2 = 2,097,150$

6. The seven generic domains are:

Commercial organizations (com)

Educational institutions (edu)

U.S. governmental organizations (gov)

International organizations (int)

U.S. military (mil)

Networks (net)

Other organizations (org)

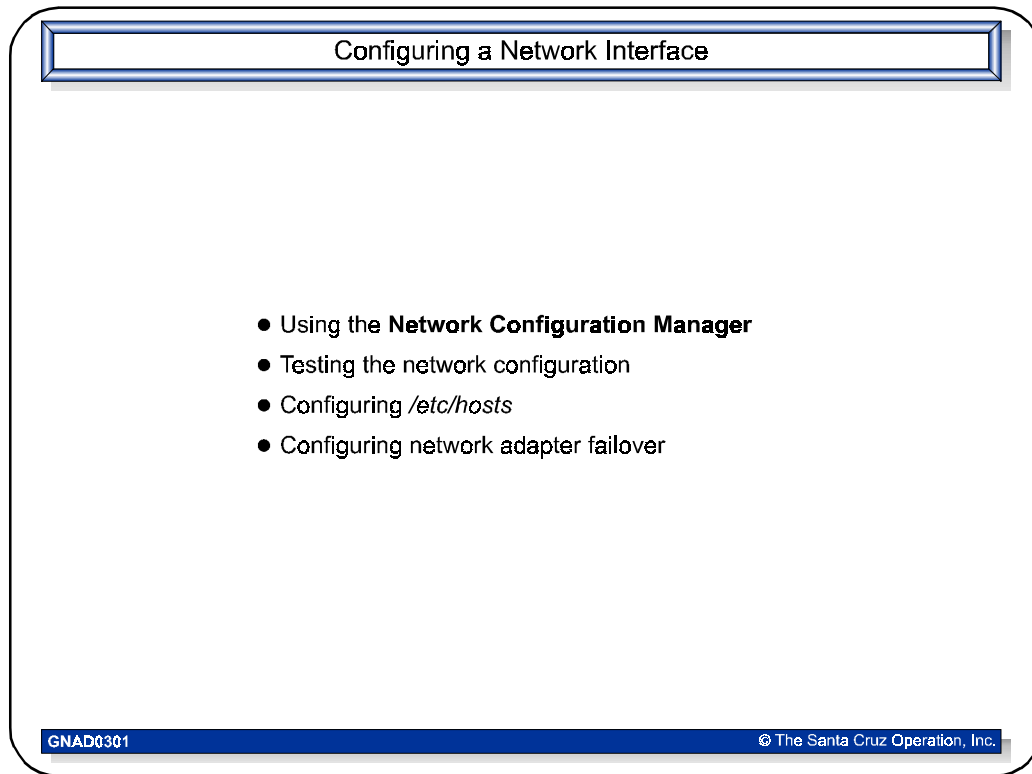
7. A router is a special purpose multi-homed host that can pass messages to another network.

8. The address resolution protocol (ARP) translates IP addresses to hardware addresses. The requesting host broadcasts a message to all hosts on the local network. This message requests that the owner of the IP address in question respond. The host that owns the IP address responds with its hardware address whereas the other hosts ignore the request. An ARP cache is also used to increase the efficiency of this sequence.
9. The five layers of the TCP/IP protocol stack include:
 - Application layer
 - Transport layer
 - Internet layer
 - Data link layer (also called network interface layer)
 - Physical layer
10. The Transmission Control Protocol (TCP) offers connection-oriented, reliable stream service between two hosts on the network. It is the primary transport protocol used by TCP/IP applications. The User Datagram Protocol (UDP) is a simpler, connectionless transport-layer protocol used to exchange requests and replies between two networked hosts. Unlike TCP, it offers no reliability guarantees.
11. A server is a process that offers a service that can be reached over a network. It accepts requests over the network, performs its service, and then returns the result to the requester. A process becomes a client when it sends a request to a server and waits for a response.
12. The `/etc/hosts` file lists the IP addresses and host names (and alias names) of the hosts that share the network. It allows users to remember hosts by name rather than by IP address. The `/etc/inet/services` file lists the service name, port number, protocol name, and aliases of well-known network services.

3

Configuring a Network Adapter

AV 3-1



Lesson 3

Configuring a Network Adapter

Introduction

To configure network interfaces on your system, you must be familiar with the attributes of your network adapter and understand the configuration parameters for your networking protocol.

After you have successfully connected your system to a network, additional configuration of network client services might be required. One of the most common client services requiring additional configuration is host name resolution.

This lesson includes the information and tasks you will need to configure these basic networking services on your system.

Objectives

This lesson details the steps necessary to add a network adapter and configure the TCP/IP protocol using the **scoadmin(1M) Network Configuration Manager**. By the end of this lesson, you will be able to:

- Use the **Network Configuration Manager** to configure networking interfaces
- Test the networking configuration
- Map host names to IP addresses using the `/etc/hosts` file
- Configure a backup network adapter

3.1 Configuring Network Interfaces

Networking is enabled on a UnixWare 7 system by creating network interfaces that consist of compatible hardware and software components *chained* together. To add a local area network (LAN) interface, you must configure:

Hardware	Set up your network hardware using utilities supplied with your computer system or network adapters.
Adapter drivers	Add a new LAN adapter driver to work with your network adapter.
Protocols	Configure an SCO protocol stack (such as TCP/IP or IPX/SPX) to communicate on the network.

After networking hardware has been added to the system, use the **Network Configuration Manager** to create network interfaces by installing drivers and configuring protocols.

Detailed procedures for configuring network interfaces begin in Section 3.1.2, “Network adapter drivers,” on page 3-7.

3.1.1 Network hardware

UnixWare 7 supports a number of common network adapters for PCI, ISA, EISA and Plug-and-Play systems. Although many adapters can be automatically detected and configured, you might need to provide some or all of the following information to configure the network interface:

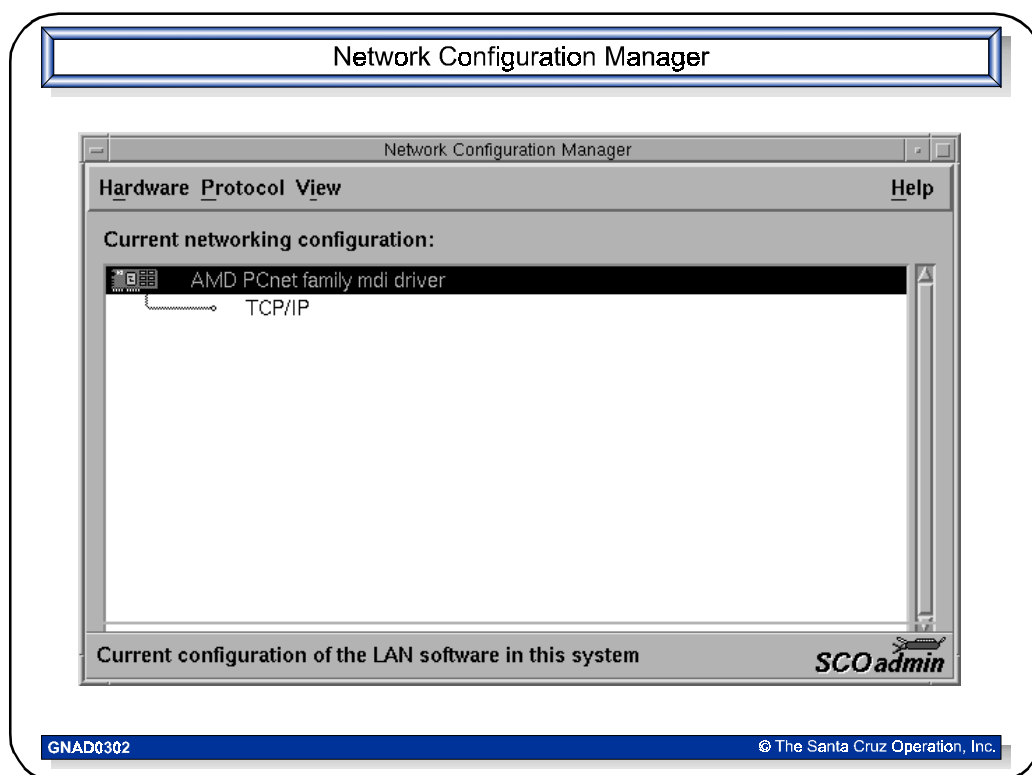
- IRQ
- I/O base address
- RAM base address
- RAM size of the network card
- DMA channel

Note that some network adapters do not use all of these parameters.

Supported network adapters are listed in the ***UnixWare 7 Getting Started Guide***. The latest list of supported hardware can be found on the Compatible Hardware Web Pages at:

<http://www.sco.com/chwp>

AV 3-2



3.1.2 Network adapter drivers

To configure a local area network (LAN) connection on your machine, you must first configure the network adapter hardware that will be used. UnixWare 7 includes drivers for network adapter cards on PCI, ISA, and EISA buses. These drivers make up the data link layer discussed in Section 2.7.1, “TCP/IP layers,” on page 2-45.

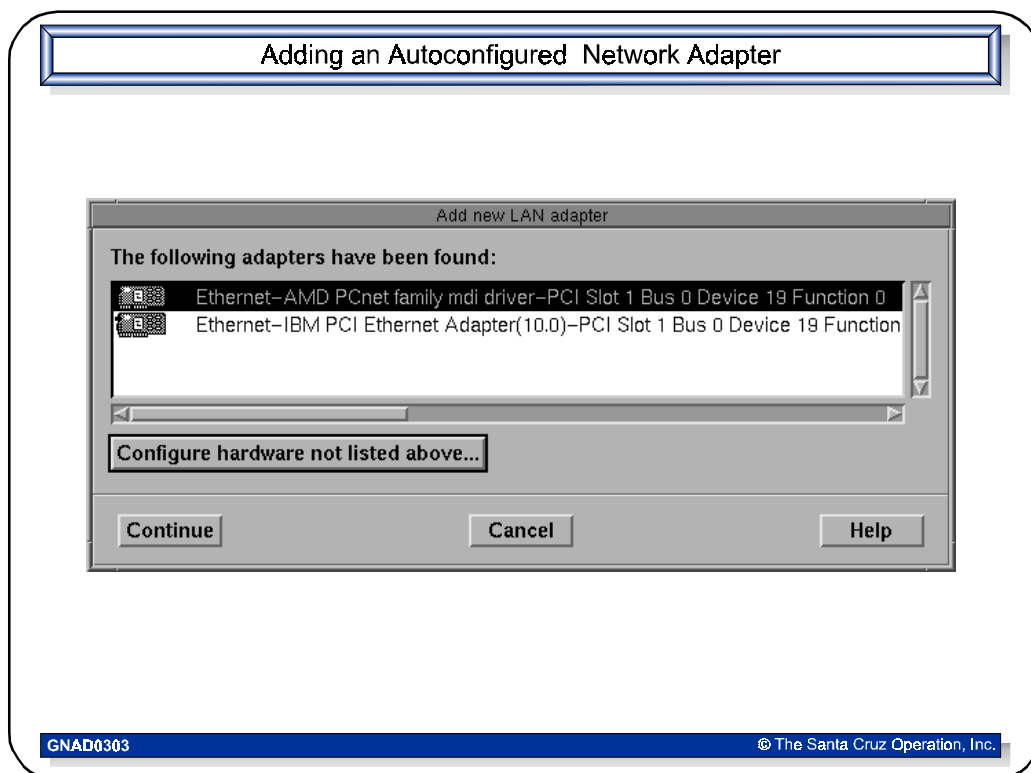
The MAC Driver Interface, MDI is the current driver technology used in UnixWare 7.

The **Network Configuration Manager** allows you to configure network adapter drivers and manage network connections. The **Network Configuration Manager** can be started in any of the following ways:

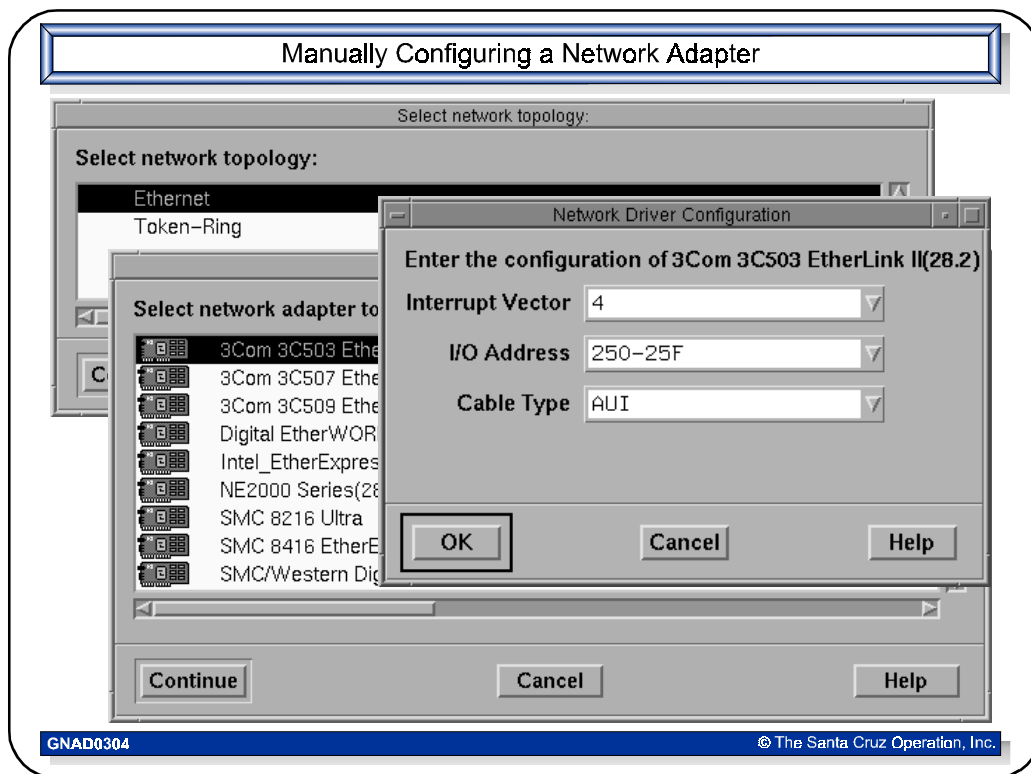
- From **SCOadmin**, select **Networking** ⇒ **Network Configuration Manager**.
- From the command line, enter **netcfg**.
- From the command line, enter **scoadmin network config**.

In the **Network Configuration Manager**, you can choose between two views for displaying information, modifying parameters, and configuring your network. For LAN connections, select **View** ⇒ **LAN**. This is the default view and is used for all procedures in this lesson.

AV 3-3



AV 3-4



3.1.2.1 Adding a network adapter

To add a network adapter:

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select **View** ⇒ **LAN**.
- Step 3. Select **Hardware** ⇒ **Add new LAN Adapter**.
- Step 4. The manager now attempts to automatically detect your network adapter.

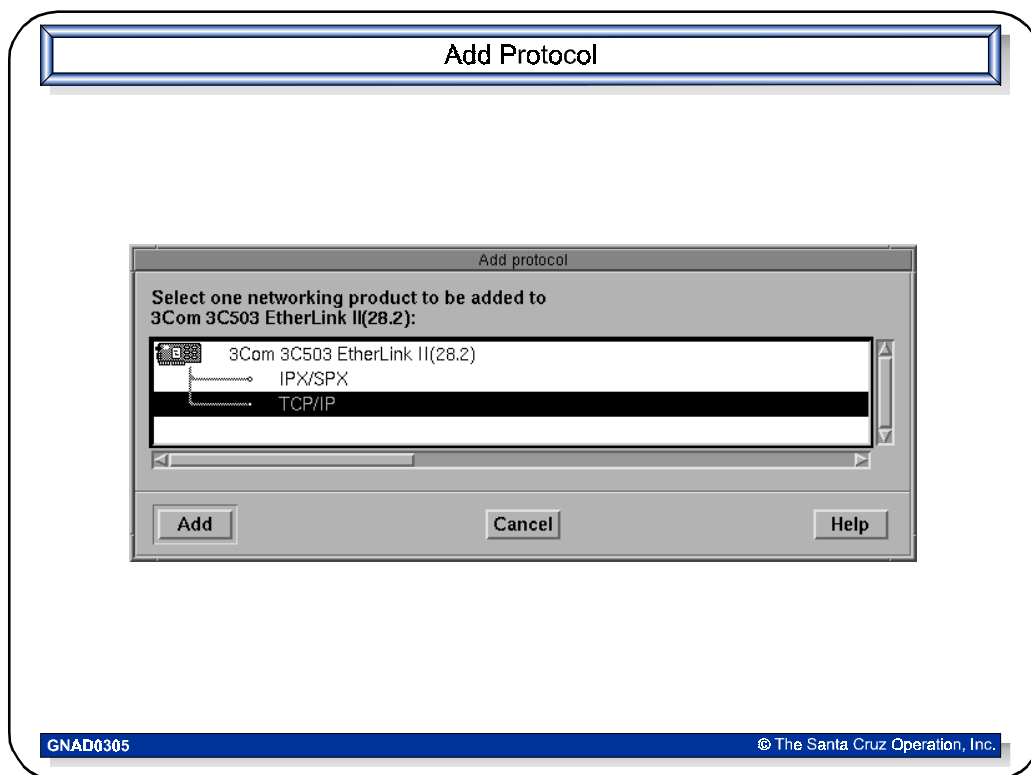
If any unconfigured adapters are detected, they are listed in a pop-up window.

- If the card you are configuring is listed, select it. Click on **Continue** and skip to Step 9.
- If the card you are configuring is not listed, click on **Configure hardware not listed above** and proceed to Step 5.

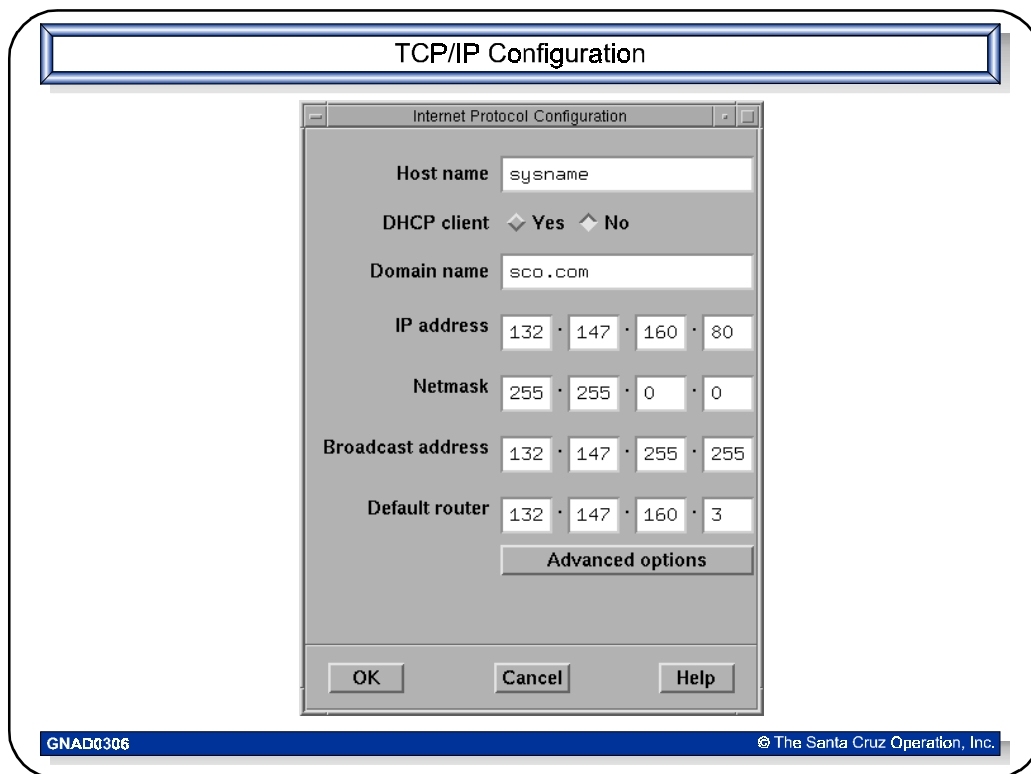
If no network adapters were detected, proceed directly to Step 5. Note that many ISA network adapters cannot be autodetected.

- Step 5. Select a network topology (for example, Ethernet). Click on **Continue**.
- Step 6. Select an adapter from the list of supported adapters (arranged by vendor), or, if prompted, choose to search for your adapter. Click on **Continue**.
- Step 7. Enter configuration information for your adapter (such as interrupt vector and I/O address) and click on **OK**. The fields are different for different models of network adapters.

AV 3-5



AV 3-6



- Step 8. Click on **OK** to verify that your card has been successfully configured.
- The **Add Protocol** window is automatically displayed when your card has been successfully configured. If you click on **Cancel** without configuring a protocol, the network adapter is not configured.
- Step 9. Select the **TCP/IP** protocol and click on **Add**.
- Step 10. Enter the appropriate values for the displayed fields when the **SCO TCP/IP Configuration** window displays.
- | | |
|-------------------|---|
| Host name | Enter the name by which this system will be known on the network. |
| Domain name | Enter the name of the domain for this system. |
| IP address | Assign an IP address to this network node. |
| Netmask | Enter the appropriate netmask. |
| Broadcast address | Enter the broadcast address. |
| Default router | If you are using a default route, enter the IP address of your router. For more information, see Section 5.2.6, "Default routes," on page 5-21. |
- Step 11. Click on **OK** when finished.
- Step 12. Click on **OK** to acknowledge the configuration of the protocol.
- Step 13. Exit from the **Network Configuration Manager**. Select **Hardware** ⇒ **Exit**.
- Step 14. Reboot the system to stop and restart TCP/IP.

3.2 Testing TCP/IP Connectivity

After configuring your TCP/IP interface, verify that your networking configuration is working properly:

- Use **netstat**(1Mtcp) to examine the currently configured interfaces.
- Try to access other systems using **ping**(1Mtcp.)

If you encounter problems contacting other systems on the network:

- Use the *loopback* interface to check the configuration of the software layers.
- Check the hardware configuration.

3.2.1 The **netstat(1Mtcp)** command

The **netstat(1Mtcp)** command is used to display configured network interfaces and other network-related data structures. The commonly used syntax is:

```
netstat -i [-n]
```

where:

- i** Shows the state of interfaces that have been auto-configured
- n** Shows network addresses with numbers (IP addresses) instead of hostnames. This prevents **netstat** from using name resolution. If there are problems with the network, name resolution might not be available.

The **netstat** output displays the *loopback* interface, *lo0*, plus one interface for each networking card or serial line interface configured on your system. If no interfaces are displayed, TCP/IP has not been configured correctly. Refer to troubleshooting information in **scohelp** ⇒ **Networking** ⇒ **Configuring network hardware and protocols** ⇒ **Configuring LAN connections** ⇒ **Troubleshooting network configuration**.

The *loopback* interface is discussed in Section 3.2.3 on page 3-15.

3.2.2 The ping(1Mtcp) command

The **ping** command verifies that the TCP/IP stack is configured correctly by sending packets to the desired destination. If the remote system has been contacted successfully, it responds by returning the packets.

The basic syntax is:

```
ping [-options] hostname
ping [-options] IP_address
```

Verify your network connections by using **ping(1Mtcp)** to contact one or more hosts on the local network that are running TCP/IP:

```
# ping cousin
cousin is alive
```

```
# ping 132.147.193.67
no answer from 132.147.196.67
```

One of the commonly used options to **ping** is **-s**. This sends one datagram per second, prints one line of output for every response that it receives, and calculates packet loss statistics. Press the interrupt key (often  or  c) to stop sending packets.

3.2.3 The *loopback* interface

If you have trouble contacting other systems on the network, verify that the software layers of the protocol stack are functioning using the special *loopback* interface.

The *loopback* interface is automatically added when TCP/IP is installed and is used to handle communication between TCP/IP client and server processes running on the same system. X Windows clients, such as **scohelp**(X1), are examples of applications which use TCP/IP client processes.

The *loopback* interface is also useful for testing and debugging the TCP/IP protocol stack because of several unique attributes:

- The *loopback* interface isolates the software layers of the protocol stack (Application, Transport, and Internet layers); hardware problems will not affect the *loopback* interface.
- No data is transmitted onto the physical network when using the *loopback* interface.
- A special host name, *localhost*, is used with the *loopback* interface as an alias for your local host.

Use *localhost* in conjunction with TCP/IP utilities such as **ping**(1Mtcp) to test the software layers of the protocol:

ping localhost

This routes packets through the *loopback* mechanism, and should always succeed regardless of the state of your networking hardware or cabling.

Exercise: Using ping(1Mtcp) and netstat(1Mtcp)

Use **netstat**(1Mtcp) to determine the number of packets that have been transmitted on the *lo0* interface (Opkts).

Enter **ping -s localhost** to test the TCP/IP software. After you stop **ping**, enter **netstat -i** to determine the number of packets transmitted on the *lo0* interface. Did the number of transmitted packets increase?

3.2.4 Verifying network hardware

If you are unable to contact other systems on the network and you have verified that the software layers of the protocol stack are functioning (Section 3.2.3 on page 3-15), check the hardware to solve any remaining problems.

The hardware components of the TCP/IP protocol are the bottom two levels of the TCP/IP protocol stack:

- Data link layer (also called link level)
- Physical layer

Any communication to another host on the network is transmitted on the *physical* network interface, for example, *net0*, and will not cause any data to be transmitted to the *loopback* interface.

If **ping** fails when attempting to contact a remote host, use **netstat(1Mtcp)** to check the data structures for the network interface. Use these guidelines to help identify the problem:

- **Opkts** and **Ipkts** equal to zero indicates that your network adapter is probably using a different interrupt than the device driver.
- **Ipkts** is non-zero, but **Opkts** equals **0errs** indicates that the I/O address for the network card might be incorrect.
- **Ipkts** is zero and **Opkts** equals **0errs** indicates that the network cable might have a faulty connection.

Check the hardware configuration of the network interface in one of these ways:

- During system boot, configured system hardware is displayed. This is logged in */var/adm/log/osmlog* and can be viewed using the **System Logs Manager**.
- Use the **dcu(1M)** command to view the present hardware configuration.

If the network interface appears to be all right, then verify the network media operation:

- Check the cable for loose connections, frayed insulation, or kinks.
- Use a cable scanner to check the wires and connections.

The **ndstat**(1M) command might also be useful to troubleshoot problems with your network interface.

The **ndstat** utility is similar to **netstat**; it also displays usage statistics for each configured network adapter driver, including:

- Driver name
- Hardware address
- Information it receives from polling the DLPI (Data Link Provider Interface) module for information on the services it provides:
 - MAC (Media Access Control) frame transmission information
 - Media-specific services for Ethernet and Token-Ring, including the Token-Ring source routing table
 - SNAP (Sub Network Access Protocol) information, used when Ethernet-oriented stacks (such as TCP/IP) run on non-Ethernet networks

For more information about using this command and the statistics it displays, see the **ndstat**(1M) manual page.

Exercise: Using the dcu(1M) to view hardware configuration

Use the **dcu(1M)** to view the present network adapter configuration:

- Step 1. Enter **dcu** from the command line.
- Step 2. Select **Hardware Device Configuration**.
- Step 3. Find the entry for the network adapter and confirm the hardware parameters.
- Step 4. Press the **u** key.
- Step 5. Select **Exit DCU and Cancel Changes**.

3.3 Configuring Network Interface Components

You might need to modify your network configuration to:

- Add additional protocols
- Modify protocol configuration
- Remove a protocol
- Modify the hardware configuration of a network adapter
- Remove a network adapter

3.3.1 Adding protocols

After the initial configuration of a network adapter, you can add additional protocols, such as IPX/SPX.

To add an additional protocol to be used with a network adapter:

- Step 1. Start the SCOadmin **Network Configuration Manager**.
- Step 2. Verify that you are in the correct view; if uncertain, select **View** ⇒ **LAN**.
- Step 3. Select the network adapter you want to configure.
- Step 4. Select **Protocol** ⇒ **Add Protocol**
- Step 5. Select the new protocol from the Add Protocol window.
- Step 6. Click on **Add**.
- Step 7. Specify the protocol parameters.

Each additional protocol requires different configuration information.
- Step 8. Click on **OK** to complete the configuration.

Refer to **scohelp** ⇒ **Networking** ⇒ **Configuring LAN connections** ⇒ **Configuring protocols** for more information.

3.3.2 Modifying and removing components

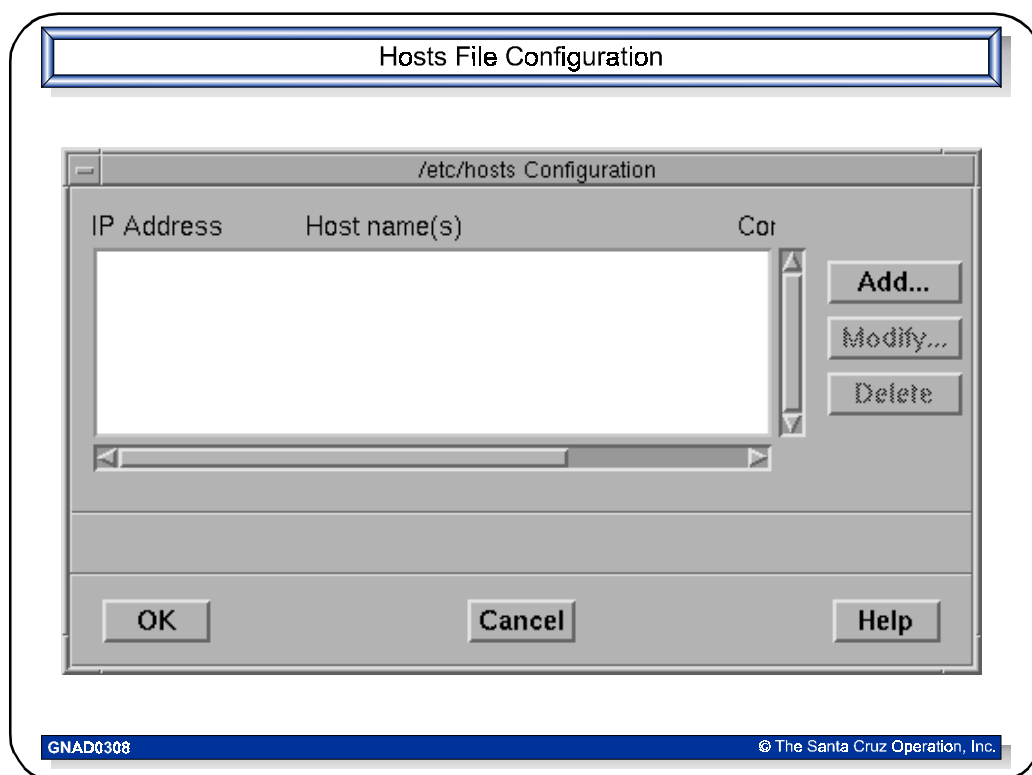
Removal and reconfiguration of network interfaces is sometimes necessary to correct problems or accommodate changes to hardware or software. Components of a chain can be individually modified or removed.

NOTE: Removing the network adapter component removes the entire chain.

To modify or remove a component:

- Step 1. Select the component in the **Network Configuration Manager** window.
- Step 2. Select one of the following:
 - **Hardware** ⇨ **Modify hardware configuration**
 - **Hardware** ⇨ **Remove network device**
 - **Protocol** ⇨ **Modify protocol configuration**
 - **Protocol** ⇨ **Remove protocol**
- Step 3. Change the appropriate information in the windows that follow if you are modifying a component.
- Step 4. Click on **OK** to complete the configuration.
- Step 5. Reboot your system to restart TCP/IP if needed.

AV 3-8



3.4 Configuring the */etc/hosts* File

After your network interface is properly configured and your system has a working connection to a network, configure the system to translate IP addresses into host names to simplify using your network.

A solution for name resolution service for a small network might be to keep a table matching IP address to hostnames on each computer. This is done using the */etc/hosts* file, which was described in Section 2.4.3, “The */etc/hosts* file,” on page 2-29.

However, this method does not scale well as the number of systems on the network increases. With a network consisting of just 20 hosts, adding a single machine would require modifying */etc/hosts* files on 21 machines. On larger networks, DNS is commonly used to provide name resolution and each system on the network is configured as a DNS client. DNS server and client configuration are covered in detail in Lesson 8: “Domain Name System (DNS) Configuration.”

Even if your system is using alternative name resolution services such as DNS, the */etc/hosts* file can be useful:

- Most systems have a small */etc/hosts* file that is used as a backup when other services are not available, such as at boot time. This usually contains entries for your host, *localhost*, and any other important hosts.
- The */etc/hosts* file can contain locally known machines or locally used aliases that are not known by the network name server.

The */etc/hosts* file can be modified using a text editor or using the **Client Manager**.

To modify the file using the **Client Manager**, highlight its entry and select **Services** ⇨ **Modify**.

3.5 Configuring Network Adapter Failover

Failover is an MDI-driver feature in UnixWare 7.1 that allows networking traffic to be shifted to a different configured network adapter card when hardware failures are detected on the original card.

If you have two network adapters that are failover-capable, you can configure your system to use one as a backup in case the primary card develops problems.

Table 2 shows the drivers supporting IP failover on UnixWare 7.1:

Table 2: Failover-capable drivers

Driver	Description
cnet/cnet-eisa	Compaq Netflex-3 series
d21x/d21x10	Intel 2104x based 10 Mbps Ethernet Controller
d21x/d21x100	Intel 2114x based 10/100 Mbps Ethernet Controller
e3H/e3H-pci	3Com EtherLink XL
eeE/eeE	Intel(R) PRO/100B / PRO/100+
hfm/hfm	HP 10/100VG PCI LAN
shrk/osi2300	Osicom 2300
shrk/shrk	Matrox Shark NS-100/4
shrk/snapper	Matrox Snapper NS-100/1
skfp/skfp99	SysKonnnect FDDI PCI Adapter
trps/trps-pci	IBM Token-Ring PCI Family Adapter

Failovers occur in one of three ways, depending on the capabilities of the network adapter:

- The driver for determines the card isn't working and sends a "failover" message up the protocol stack, and the stack starts using the backup device automatically.
- The protocol stack notices a backlog of data headed to a failover-capable network adapter (with backup) and sends a probe packet to the card. If the probe fails in any way, the protocol stack causes a failover and automatically starts using the backup device.
- An administrator determines the network adapter is not working working correctly (typically because of a call from a user) and forces a failover.

To configure failover network adapters:

- Step 1. Use the **Network Configuration Manager** to configure the first failover-capable network adapter as described in Section 3.1.2.1, "Adding a network adapter," on page 3-9.
- Step 2. Configure a second failover-capable network adapter. When the protocol list is shown, select "Backup" as the protocol.
- Step 3. Indicate which network adapter this new adapter will back up. Select the network adapter configured in Step 1.

To force a failover:

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select the primary network adapter
- Step 3. Select **Hardware** ⇌ **Switch to backup**

The **Network Configuration Manager** allows you to force a *failback*, which moves traffic back to the original card after the problem is resolved. This is useful when the backup network adapter card is not as good as the primary, such as a 100 Mb network card that is backed up by a 10 Mb card. Under this circumstance, you want to get traffic back on the primary network card as soon as possible.

To reuse the primary network adapter:

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select the backup network adapter.
- Step 3. Select **Hardware** ⇌ **Reuse Primary**.

To bypass the software components and test whether a network adapter is really working:

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select the network adapter.
- Step 3. Select **Hardware** ⇨ **Test Network Connectivity**.

If the last line says "success", a packet has been loaded onto the card, sent out the cable, and read back again. Thus, the hardware is working.

3.6 Lesson Summary

This lesson described the steps necessary to configure a network adapter on a system and to set up the protocols and client services necessary to connect to a LAN or to the Internet. Some of the necessary concepts and tools discussed include:

- Hardware attributes of network adapters, including DMA, interrupts, I/O addresses, and memory-mapped I/O
- Using the **Network Configuration Manager** to configure drivers and protocols
- Configuring TCP/IP
- Testing the connection using the **ping**(1Mtcp) and **netstat**(1Mtcp) commands
- The *loopback* interface
- Configuring the */etc/hosts* file
- Configuring network adapter failover

Review questions

1. What are four network adapter hardware attributes that you might need to configure?
2. What is the purpose of the **Network Configuration Manager**?
3. What is the purpose of the *loopback* interface? How can you use this interface to test the software configuration?

4. Which two entries are usually present in every */etc/hosts* file?

5. What is the purpose of the **netstat**(1Mtcp) command?

3.7 Lab: Configuring a Network Node

In this two-part lab exercise, you will first configure your network adapter and TCP/IP using the information provided by your instructor. After you have verified that your network interface is working, you will configure DNS client services to establish a network with the other systems in your classroom.

3.7.1 Configuring a network adapter

You will use the **Network Configuration Manager** to add a chain to your system. This will require the addition of a network adapter and configuration of the TCP/IP protocol. Your instructor will provide you with the following items (some of which might not apply for your particular hardware setup):

- Name of network adapter
- IRQ
- I/O addresses
- RAM base address for memory-mapped I/O
- RAM size
- IP address

You need to know the netmask and broadcast address to complete this configuration. You can determine these values from the IP address provided (you will not be subnetting this particular network).

NOTE: Before attempting to configure the network adapter, it might be necessary to first remove the existing configuration if this adapter was configured prior to the start of class. Your instructor will tell you if this is necessary.

Procedure

If directed by the instructor, first remove the existing network adapter configuration:

- Step 1. Run **scoadmin** ⇒ **Networking** ⇒ **Network Configuration Manager**.
- Step 2. Remove the existing network adapter configuration if necessary.
- Step 3. Select the network adapter from the **Network Configuration Manager** window.
- Step 4. Select **Hardware** ⇒ **Remove network device**.
- Step 5. Click on **Yes** to confirm removal of the network devices.
- Step 6. Click on **OK** to acknowledge the removal of the device.

After the network adapter is removed, configure a new network adapter:

- Step 1. Run **scoadmin** ⇒ **Networking** ⇒ **Network Configuration Manager**.
- Step 2. Select **Hardware** ⇒ **Add new LAN adapter**.
- Step 3. The manager now attempts to autodetect your network adapter.

If any unconfigured adapters are autodetected, they are listed in a pop-up window.

- If the card you are configuring is listed, select it. Click on **Continue** and skip to step 8.
- If the card you are configuring is not listed, click on **Configure hardware not listed above** and proceed to step 4.

If no network adapters were autodetected, the list does not appear and you proceed directly to step 4.

Note: Many ISA network adapters cannot be autodetected.

- Step 4. Select a network topology (for example, Ethernet) and click on **Continue**.
- Step 5. Select an adapter from the list of supported adapters (arranged by vendor), or, if prompted, choose to search for your adapter. Click on **Continue**.

Step 6. Enter configuration information for your adapter (such as interrupt vector, I/O address) and click on **OK**. The fields are different for different models of network adapters.

Step 7. Click on **OK** to confirm that your card has been successfully configured.

The **Add Protocol** window automatically displays when your card is successfully configured. If you click on **Cancel** without configuring a protocol, the network adapter is not configured.

Next, configure the TCP/IP protocol:

Step 8. Select **TCP/IP** in the Internet Protocol Configuration window and click on **Add**. This opens the window.

Step 9. Enter the host's name in the "Host name" field.

Step 10. Enter the name of the domain to which this machine belongs in the "Domain name" field.

Step 11. Enter the IP address, netmask, and broadcast address of this network adapter in the "IP Address", "Netmask", and "Broadcast address" fields.

Step 12. Enter the default router, if you are using one, in the "Default router" field.

Step 13. Click on **OK** at the bottom of the window. This displays a window informing you that the network adapter was successfully configured.

Step 14. Click on **OK** at the bottom of the next window to complete the configuration.

Step 15. Exit the **Network Configuration Manager**. Reboot the system.

The network adapter is now configured and ready to test:

Step 16. Check your connections with the **ping(1Mtcp)** command using IP addresses. Use your own address as well as others' in the classroom.

Step 17. Use the **netstat(1Mtcp)** command from a terminal window to examine the currently configured interfaces and other network-related data structures.

3.8 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

dcu (1M)	Device Configuration Utility
finger (1tcp)	Display information about local and remote users
ndstat (1M)	Displays network adapter driver statistics
netcfg (1M)	Network configuration utility which can be invoked from the command line, or from scoadmin (1M)
netstat (1Mtcp)	Displays the contents of various network-related data structures
ping (1Mtcp)	Tests network connections by eliciting hosts to reply to packets sent to them

Relevant files

<i>/etc/hosts</i>	Local table for hostname to IP address matching. See <i>hosts</i> (4tcp).
<i>/etc/netconfig</i>	The network configuration database, a system file used to store information about networks that are connected to the system and that are available for use. See <i>netconfig</i> (4bnu).
<i>/etc/resolv.conf</i>	Name resolution configuration file. See <i>resolv.conf</i> (4tcp).

3.9 Answers to Review Questions

1. The network adapter hardware attributes you might need to configure include Direct Memory Access (DMA), interrupts (IRQ), I/O addresses, and memory-mapped I/O addresses.
2. The **Network Configuration Manager** allows you to add, modify, view, and remove network interface configurations.
3. The *loopback* interface is a special network interface that allows client and server processes on a single system to communicate locally using TCP/IP. You can use this interface to test your system's TCP/IP protocol stack setup without transmitting data to the physical network. For example, use the **ping(1Mtcp)** command to test the software configuration. Enter: **ping localhost**.
4. Host files should contain entries for *localhost* and the local machine name.
5. The **netstat(1Mtcp)** command allows you to examine the currently configured interfaces and other network-related data structures.

4

Subnets

AV 4-1

Subnetworks

- Planning the subnet topology
- Defining subnet IP addresses, netmasks, and broadcast addresses

GNAD0401 © The Santa Cruz Operation, Inc.

Lesson 4

Subnets

Introduction

Recall that a Class A network can have more than 16 million nodes (hosts), a Class B network can have 65,534, and a Class C network can have 254. Managing the delivery of data to all these nodes is very difficult (or impossible) if all of them are connected to a single network segment.

Because it is not always feasible to connect all nodes to the same physical network, networks are sometimes divided into subnets. Networks are also divided into subnets for reasons of performance, ease of administration, security, and territory.

This lesson helps you plan a network topology that fixes pre-existing problems, accommodates your present needs, and allows for future expansion.

Objectives

This lesson discusses how to implement subnets. By the end of this lesson, you will be able to:

- Plan a subnet topology based upon your constraints and performance needs
- Define netmasks, network numbers, broadcast, and IP addresses for subnets

4.1 Planning a Subnet Topology

In an ideal world, your network topology would be simple:

- You would have a single official IP network address given to you.
- You could fit all of your nodes onto your chosen physical layer.
- The physical layer could span the globe without routers.
- The physical layer would be infinitely fast so performance would never be an issue.

Because these ideals often do not exist, you will probably need to split a network into one or more subnetworks. Many of the reasons for creating subnetworks are rooted in your organization's growing reliance on the network. This growth can come in several forms:

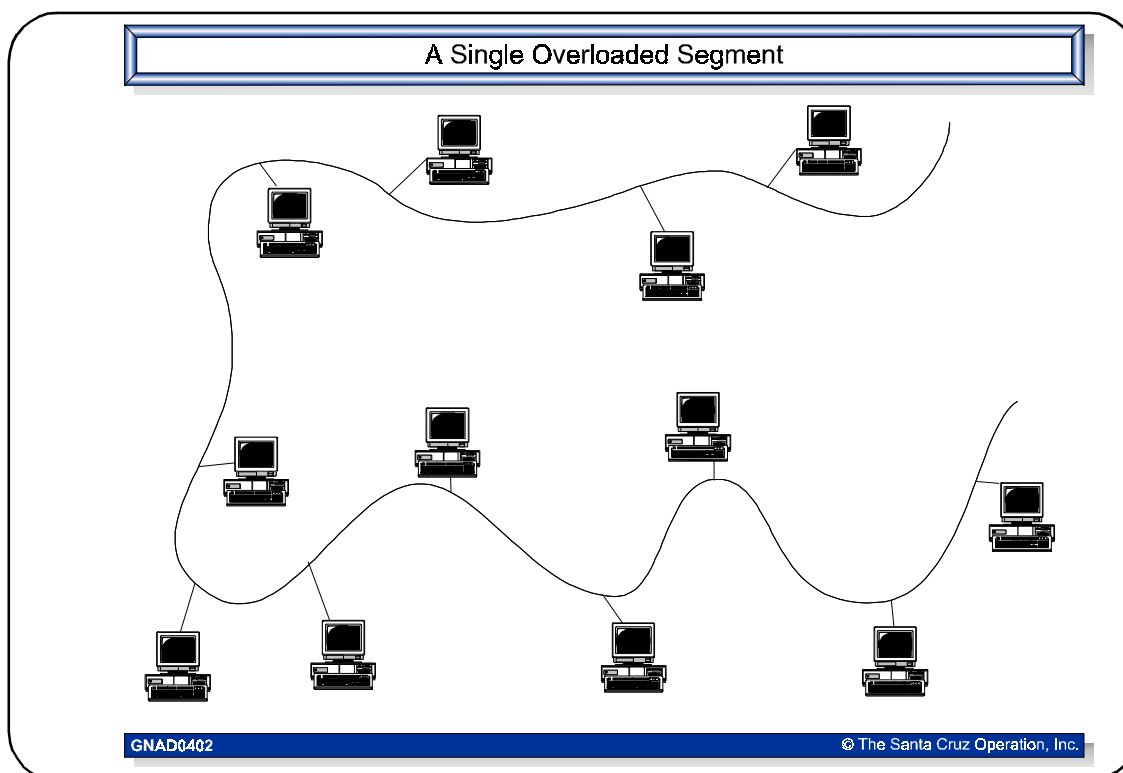
- An increase in the number of nodes
- A need to combine several independent LANs into a single enterprise LAN
- Increased traffic as services become more widely dispersed

Other advantages of creating subnetworks include:

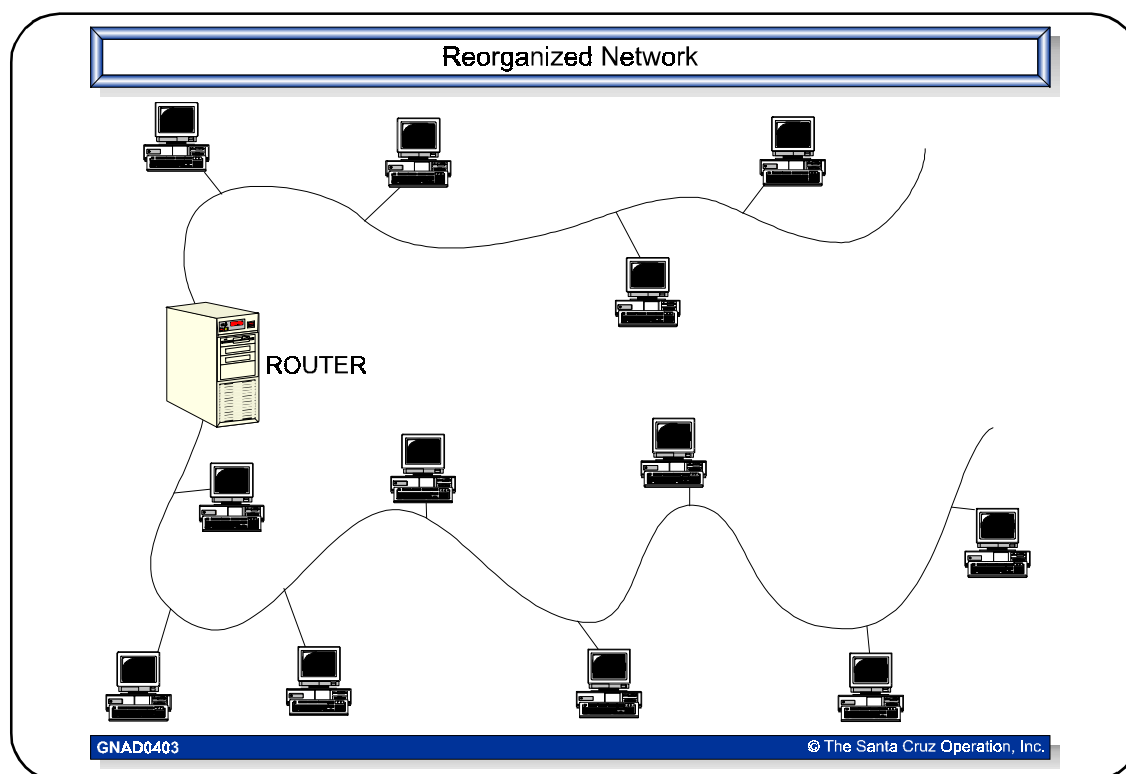
- IP address management can be distributed to local administrators.
- The network topology is hidden from the outside world; using subnets requires only a single route to external gateways.

How you decide to subnet, based on the issues you encounter, will greatly affect your job as a network administrator. The following sections discuss each of these issues in more detail and gives you guidelines for creating a plan that deals with them effectively.

AV 4-2



AV 4-3



4.1.1 Increasing the number of network nodes

As the number of nodes on a physical network increases, expansion will be limited by the type of cable in use. Each type has a limit to the number of nodes that can be attached to a single segment of cable. As your organization adds more nodes to the network, you might reach these limits, which were described in Section 2.8, “LAN Hardware,” on page 2-50.

One way to solve this problem is to split the network and add another Class C address. The problems with this approach are:

- Official Class C network addresses are almost all used and difficult to obtain.
- Adding another network to the Internet increases its routing overhead.

Another solution is to split the network into two or more subnets, as illustrated in AV 4-3 on page 4-6.

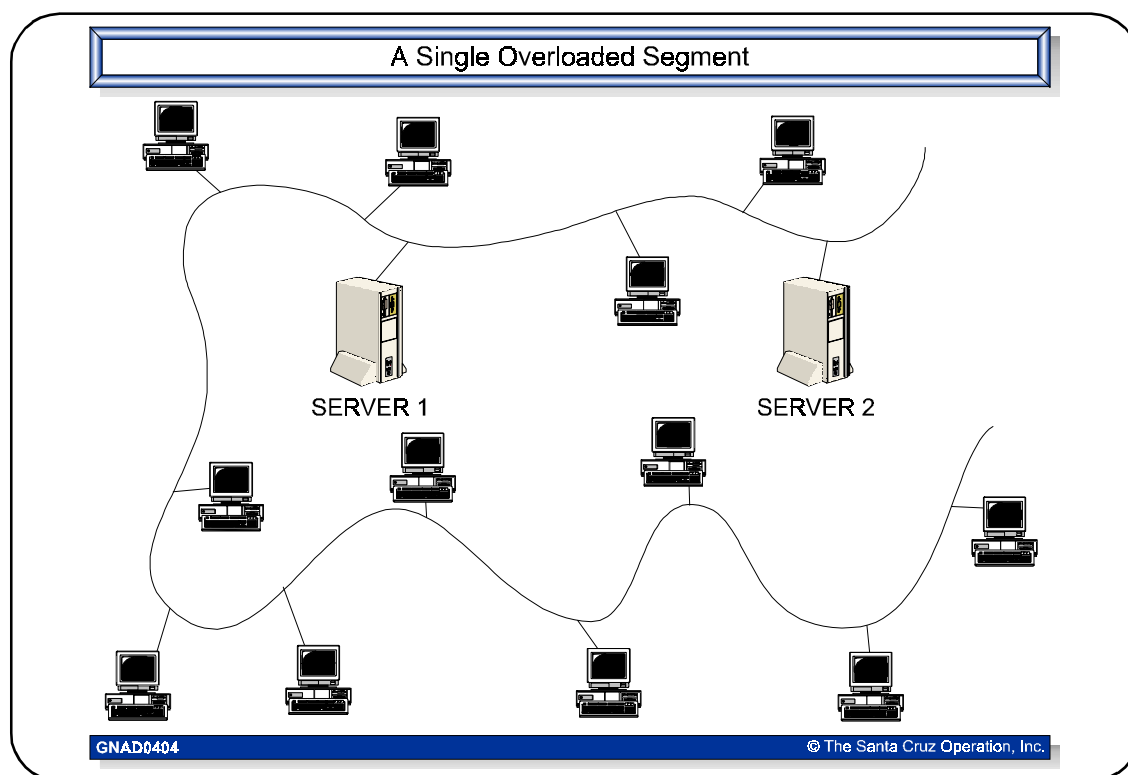
4.1.2 Joining multiple LANs

Your organization might have multiple LANs that were created in separate departments or geographies and are not connected. As users take greater advantage of their local networks, they might require access to resources on other LANs.

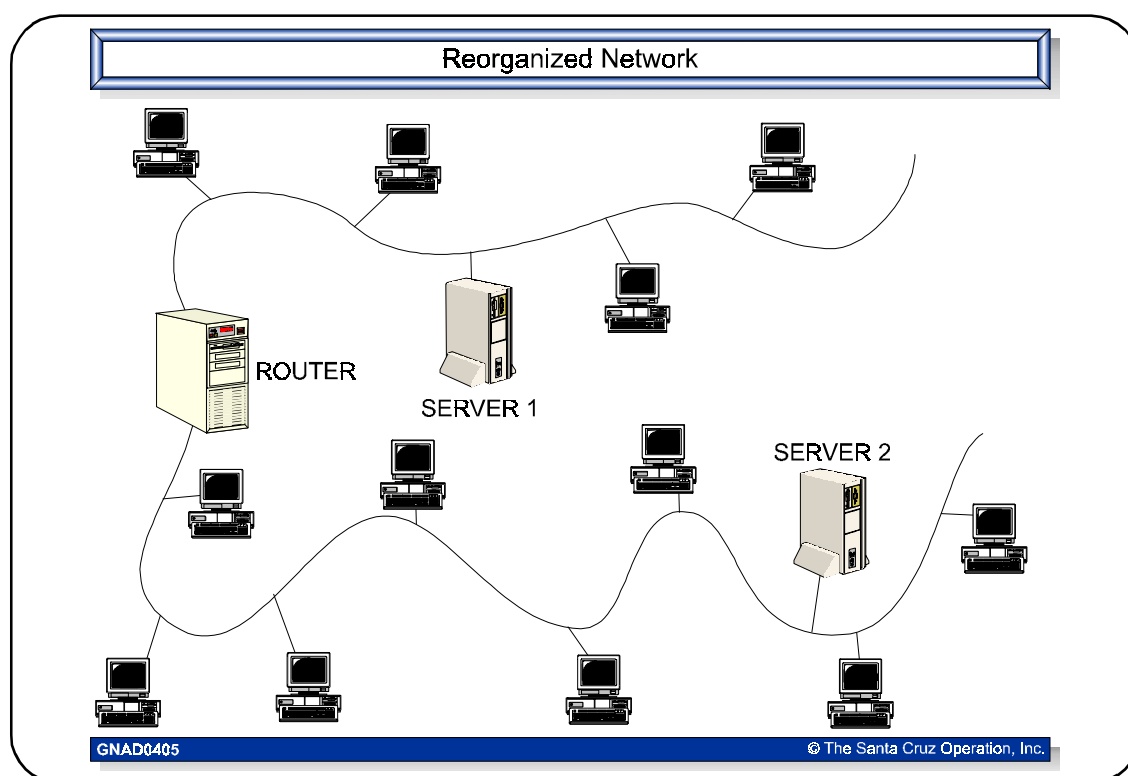
Joining all of your separate LANs into a single enterprise-wide network is the ideal solution. The problems you will face are not entirely different than when the number of nodes increases. Additionally, spanning geographic locations is usually not feasible using a LAN technology (for example, Ethernet).

Subnets provide a method of joining together separate LANs.

AV 4-4



AV 4-5



4.1.3 Increased network traffic

If you experience increased network use by existing nodes, you can use subnets to increase performance on your network:

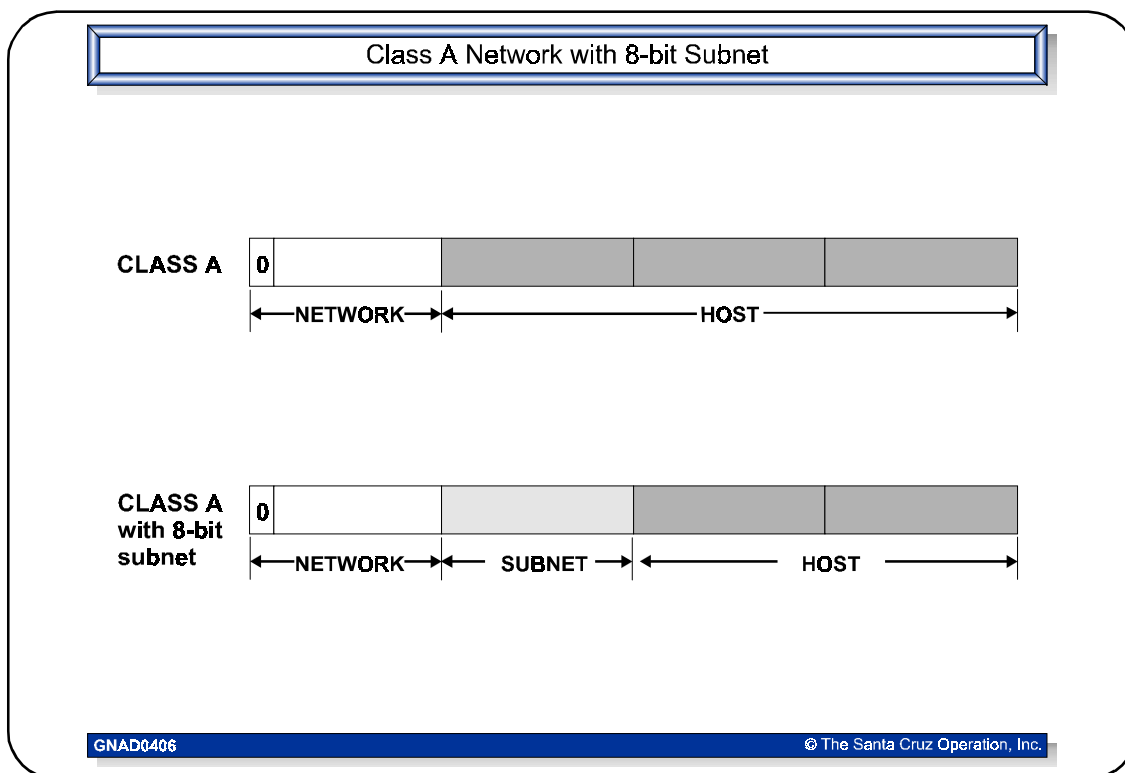
1. Identify a group of nodes that are exchanging a significant amount of traffic on the network.
2. Verify that isolating these nodes will not significantly decrease their performance.

Isolating these nodes onto their own subnet will increase performance for the remaining nodes. However, you might also decrease the performance for the moved nodes if the majority of their traffic is associated with a server on another subnet.

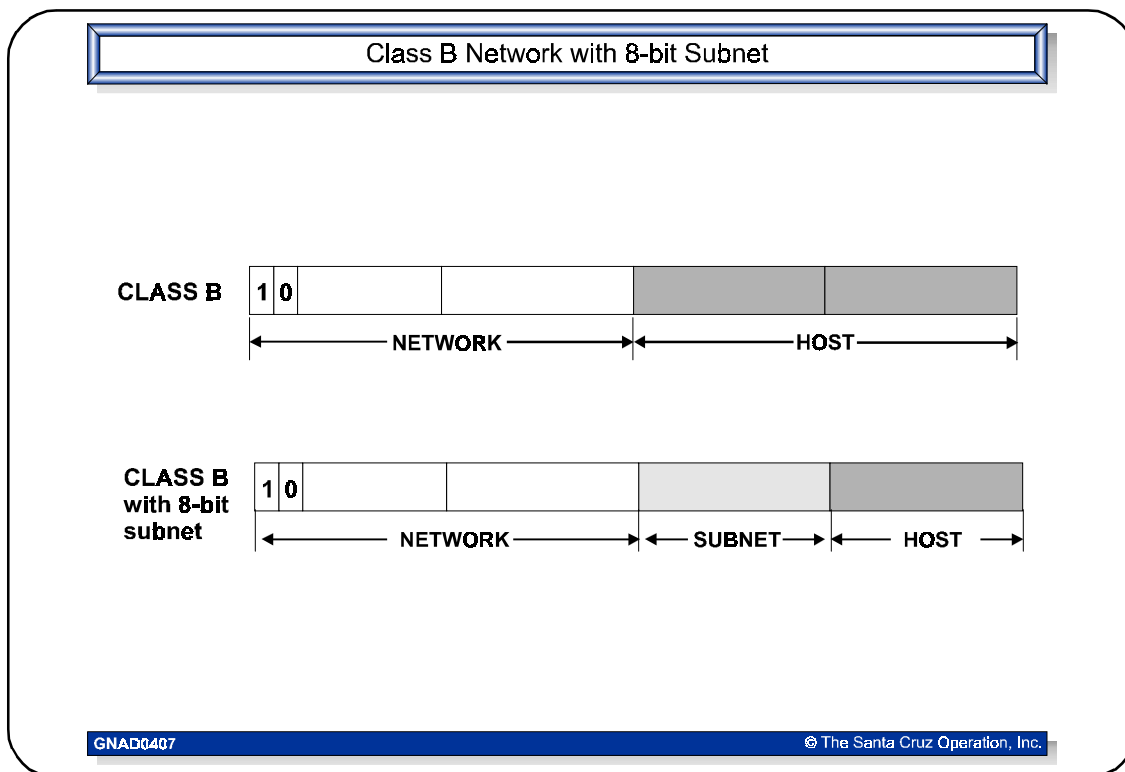
3. Isolate those machines onto their own subnet.

If network traffic is a problem, performance might also be increased by locating servers centrally so they have an interface present on all the subnets. Be careful not to overload the servers with the increased network traffic.

AV 4-6



AV 4-7



4.2 Creating Subnets

When you have determined that the network must be subdivided to solve network problems or better meet users' requirements:

- Decide how many subnets are needed.
- Determine the appropriate IP addresses for your subnets.
- Configure each subnet node with the correct IP address, netmask, and broadcast address.
- Update the host name resolution service (*/etc/hosts* or DNS, for example). Modifying the *hosts(4tcp)* file was previously discussed. Configuring and modifying DNS server entries will be discussed in Lesson 8: "Domain Name System (DNS) Configuration."
- Configure routers between subnets. Configuring routers will be discussed in Lesson 5: "Routing".

The remaining sections of this lesson provide procedures and examples for adding subnets by creating IP addresses, netmasks, and broadcast addresses for each class of network.

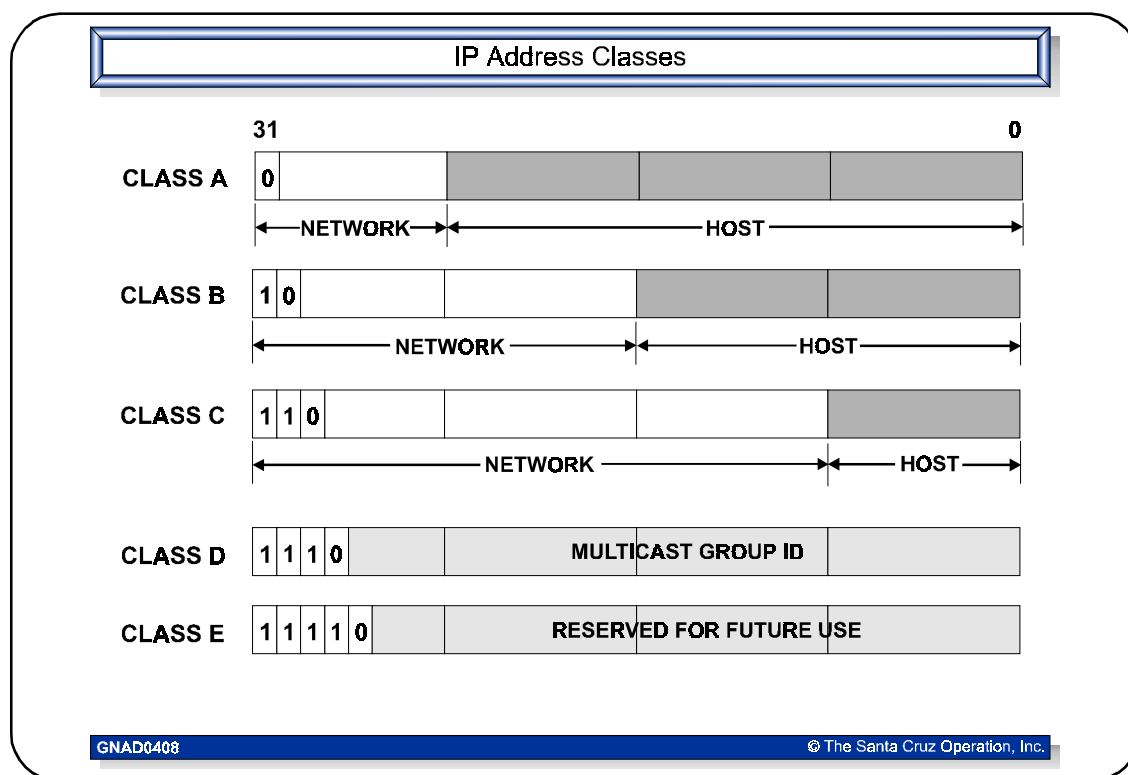
4.2.1 Dividing up the network

To decide how to partition your network, you must consider:

- The number of nodes on your network
- The number of subnets needed
- Future growth

Subnets are created by reassigning some of the bit positions of an IP address previously used to identify hosts and instead use them to identify subnets. The network class you are using will determine how you partition the network, hence the total number of subnets and nodes that can be supported.

AV 4-8



4.2.1.1 Revisiting IP addresses, netmasks, and broadcast addresses

Recall the following attributes of non-subnetted addresses:

- One part of the 32-bit (4-octet) IP address designates the network and the other part designates the host.
- The most significant (first) octet of the network portion specifies the class.
- The netmask is used to determine how many bits of the IP address are needed for the network address and how many for the host address—a one (1) indicates a network bit and zero (0) indicates a host address bit.
- The broadcast address is the network portion of the address with all host bits set to 1.

Table 3: Example IP addresses with no subnetting

Class	Most significant octet	Network bits	Host bits	Example IP addresses	Netmask	Broadcast
A	0-127	8	24	85.110.43.7	255.0.0.0	85.255.255.255
B	128–191	16	16	132.65.20.109	255.255.0.0	132.65.255.255
C	192–223	24	8	210.20.210.20	255.255.255.0	210.20.210.255

Also recall how the number of network and host bits determine the number of networks available and the number of nodes (hosts) on each network.

Table 4: Calculating the number of networks

Class	Total available networks	Nodes per network
A	$2^7 - 2 = 126$	$2^{24} - 2 = 16,777,214$
B	$2^{14} - 2 = 16,382$	$2^{16} - 2 = 65,534$
C	$2^{21} - 2 = 2,097,150$	$2^8 - 2 = 254$

Because neither the network address, the subnet address, nor the host address can have the binary value of all zeros or all ones, there are always two fewer networks and two fewer nodes than the theoretical maximum.



4.2.2 Class A subnets

Subnets are commonly used in Class A and Class B networks because of the large number of possible nodes.

A Class A network allows for more than 16 million possible nodes ($2^{24} - 2$). Because it is not feasible to support this many nodes on a single network, subnets are needed to support large networks.

The simplest way to subnet an IP address is to use one or more octets as the subnet address. A Class A network can use between 2 and 22 bits of the 24-bit host address for the subnet address, depending on the number of subnetworks and nodes needed.

4.2.2.1 Class A 8-bit subnets

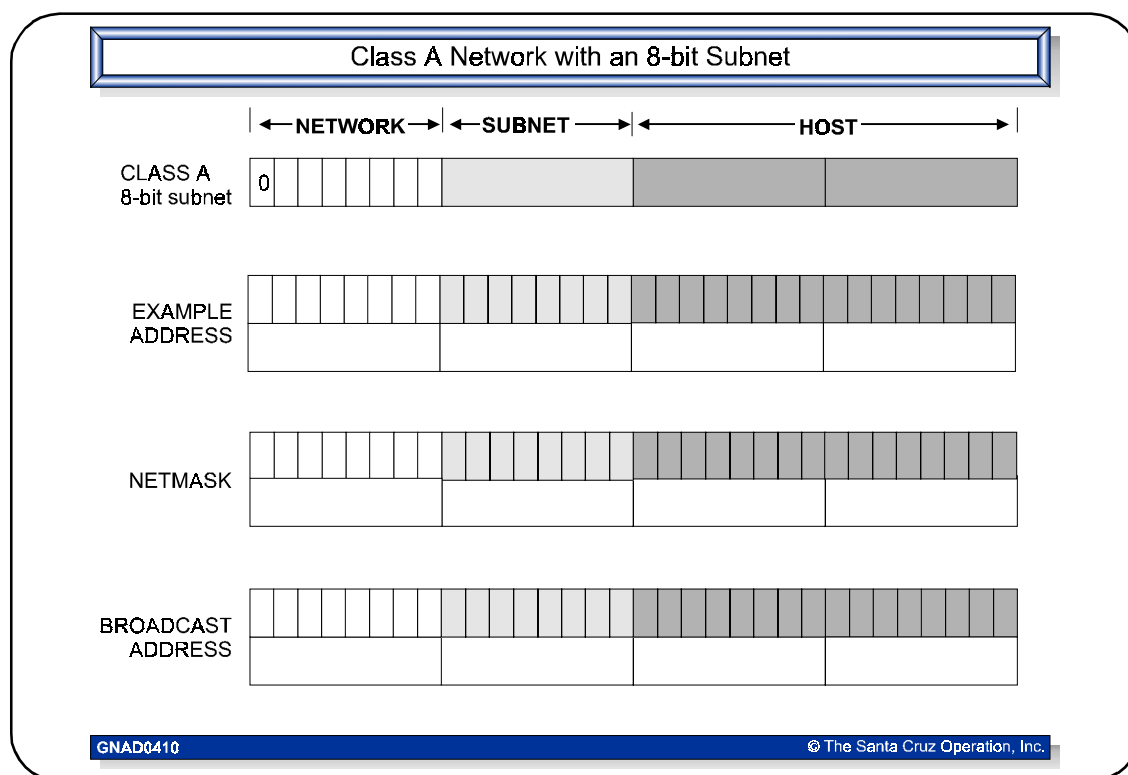
Using the second octet as the subnet portion of a Class A network address provides subnets 1 through 254, each able to support 65,534 ($2^{16} - 2$) nodes.

For example, network number 120.0.0.0 could use the second octet as a subnet address creating subnets 120.1 through 120.254.

The netmask and broadcast address are easy to calculate when subnets fall on octet boundaries. Use the following procedure and AV 4-9 on page 4-14 to find the netmask and broadcast address for a node with IP address 120.3.25.43:

- | | |
|---------|---|
| Step 1. | Enter the IP address in decimal notation: 120.3.25.43. |
| Step 2. | Convert the IP address to binary:
01111000 00000011 00011001 00101011 |
| Step 3. | Enter the netmask in binary— enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
11111111 11111111 00000000 00000000 |
| Step 4. | Convert each octet of the netmask to decimal:
255.255.0.0 |

AV 4-10



- Step 5. Enter the broadcast address in binary—enter a one in each host address bit; copy each network and subnet bit into the corresponding broadcast address fields:
01111000 00000011 11111111 11111111
- Step 6. Convert each octet of the broadcast address to decimal:
120.3.255.255

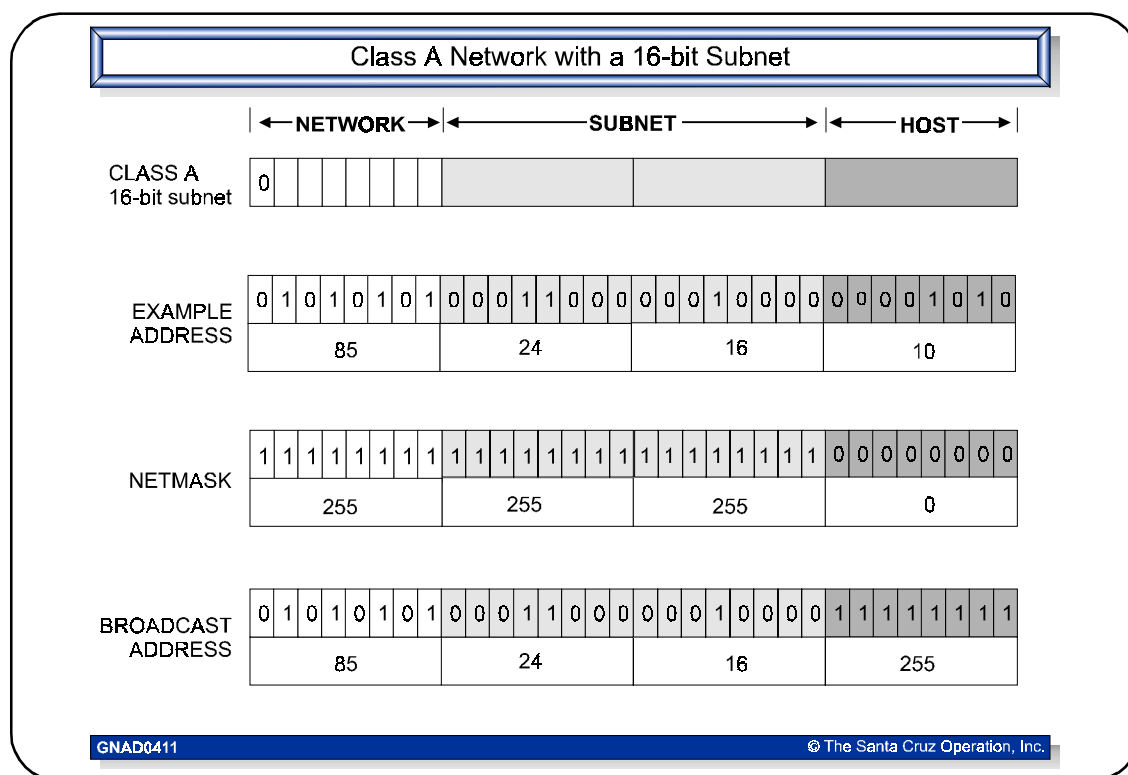
Exercise: Creating Class A 8-bit subnets

Use the form in AV 4-10 on page 4-16 to determine the netmask and broadcast address for IP address 50.75.128.125, assuming an 8-bit subnet.

Hint: Use the **bc(1)** command to do number base conversions. Indicate the output base (ob) *before* the input base (ib). For example:

```
$ bc
ob=10
ib=2
11100000
224
quit
$
```

AV 4-11



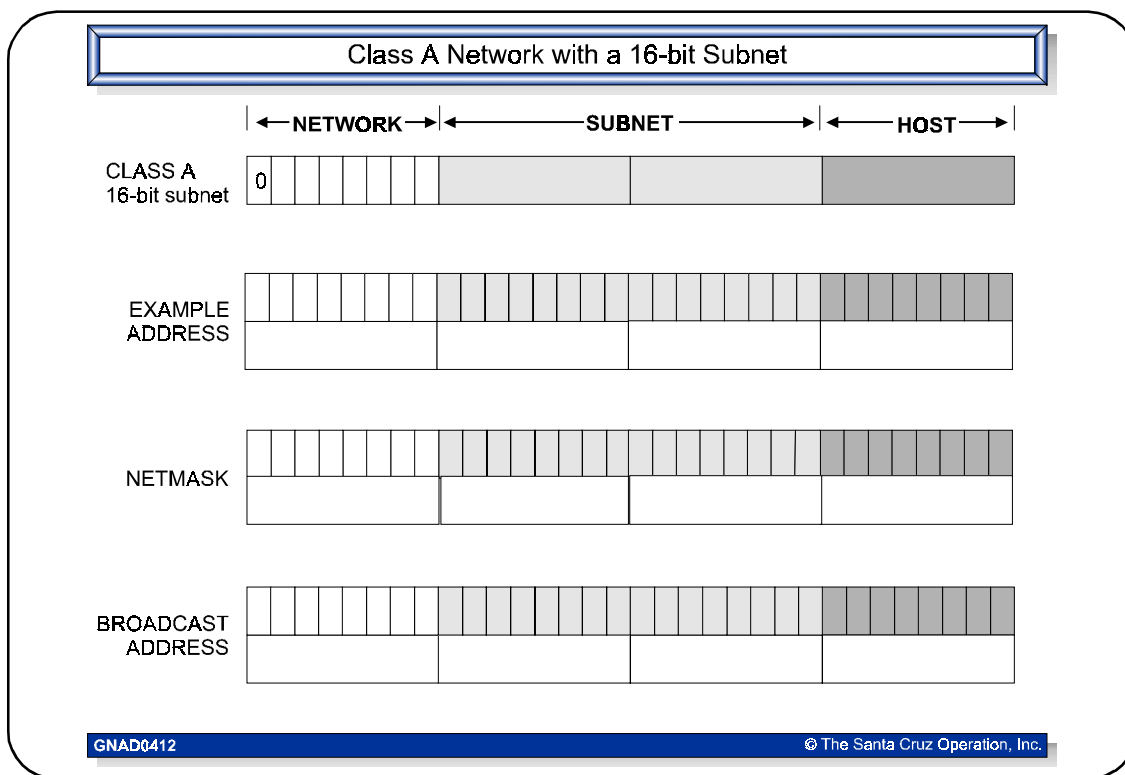
4.2.2.2 Class A 16-bit subnets

Using the second and third octets as the subnet portion of a Class A network address provides subnets 1 through 65,534 ($2^{16} - 2$) each able to support 254 ($2^8 - 2$) nodes.

For example, network number 85.0.0.0 uses the second and third octets as a subnet address creating subnets 85.0.1, 85.0.2, 85.0.3, ... 85.254.254. To determine the netmask and broadcast address for host 10 on subnet 85.24.16:

- Step 1. Enter the IP address in decimal notation: 85.24.16.10.
- Step 2. Convert the IP address to binary:
01010101 00011000 00010000 00001010
- Step 3. Enter the netmask in binary— enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
11111111 11111111 11111111 00000000
- Step 4. Convert each octet of the netmask to decimal and enter in the form:
255.255.255.0
- Step 5. Enter the broadcast address in binary—enter a one in each host address bit; next copy each network and subnet bit into the corresponding broadcast address fields:
01010101 00011000 00010000 11111111
- Step 6. Convert each octet of the broadcast address to decimal:
85.24.16.255

AV 4-12



Exercise: Creating Class A 16-bit subnets

Use the form in AV 4-12 on page 4-20 to determine the netmask and broadcast address for IP address 120.110.125.52, assuming a 16-bit subnet.



4.2.3 Class B subnets

Subnets are commonly used in Class B networks because of the large number of possible nodes.

A Class B network allows for more than 65 thousand possible nodes ($2^{16} - 2$). Because it is not feasible to support this many nodes on a single network, subnets are needed to support large networks.

The simplest way to subnet a Class B IP address is to use one octet as the subnet address. However, a Class B network can use between 2 and 14 bits of the 16-bit host address for the subnet address, depending on the number of subnetworks and nodes needed.

4.2.3.1 Class B 8-bit subnets

Class B subnets often use 8 bits (the third octet) of the host portion of the address. A Class B network which uses 8 bits as the subnet address provides 254 ($2^8 - 2$) subnets, leaving 8 bits for host addresses for a total of 254 nodes per subnet. This allows a Class B address to simulate 254 Class C addresses.

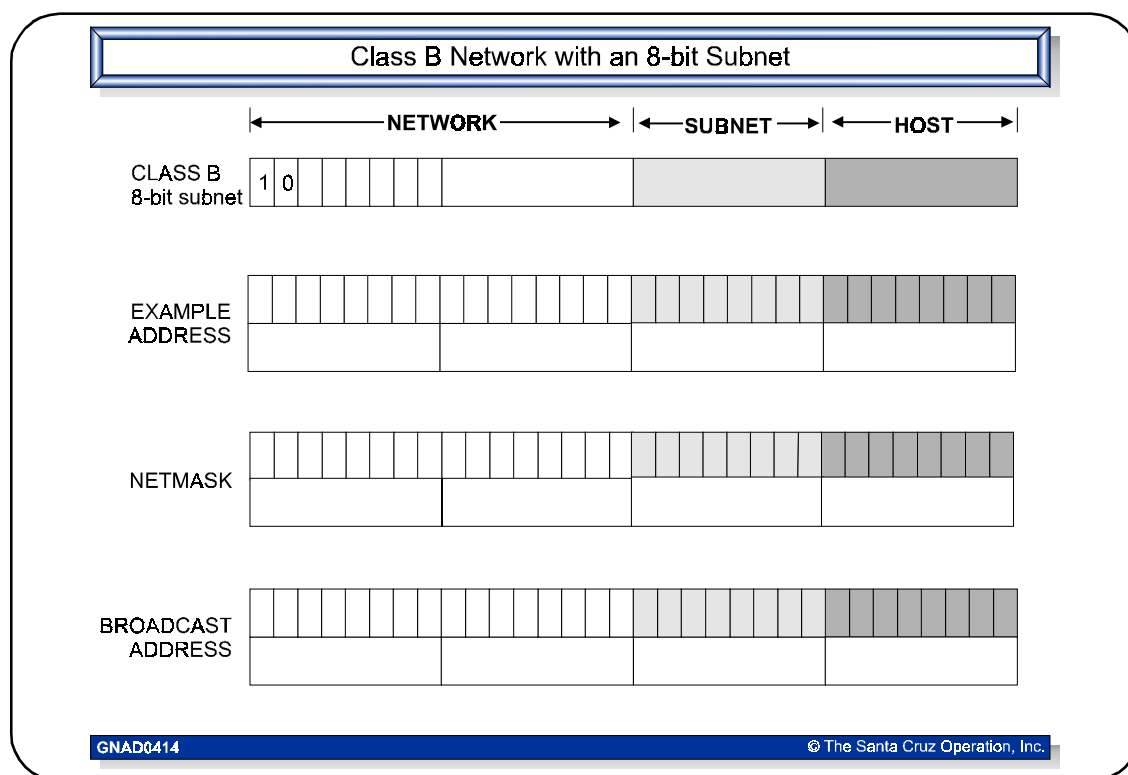
SCO is an example of an organization with a Class B address, 132.147.0.0. SCO has a large number of hosts separated by geography.

Because the hosts cannot be connected to a single network segment, 8-bit subnets are used. For example, two subnets might have the network addresses 132.147.1 and 132.147.2.

The nodes on subnet 1 have IP addresses 132.147.1.1 through 132.147.1.254. The nodes on subnet 2 have addresses 132.147.2.1 through 132.147.2.254, and so on.

Using the procedure in Section 4.2.2.2, “Class A 16-bit subnets,” on page 4-19, it is easy to determine the netmask and broadcast values for example IP address 132.147.24.138, shown in AV 4-13 on page 4-22.

AV 4-14



Exercise: Creating Class B 8-bit subnets

Use the form in AV 4-14 on page 4-24 to determine the IP address, netmask, and broadcast address for host 220 on subnet 150, assuming you are using SCO's Class B network address of 132.147.0.0 with an 8-bit subnet.

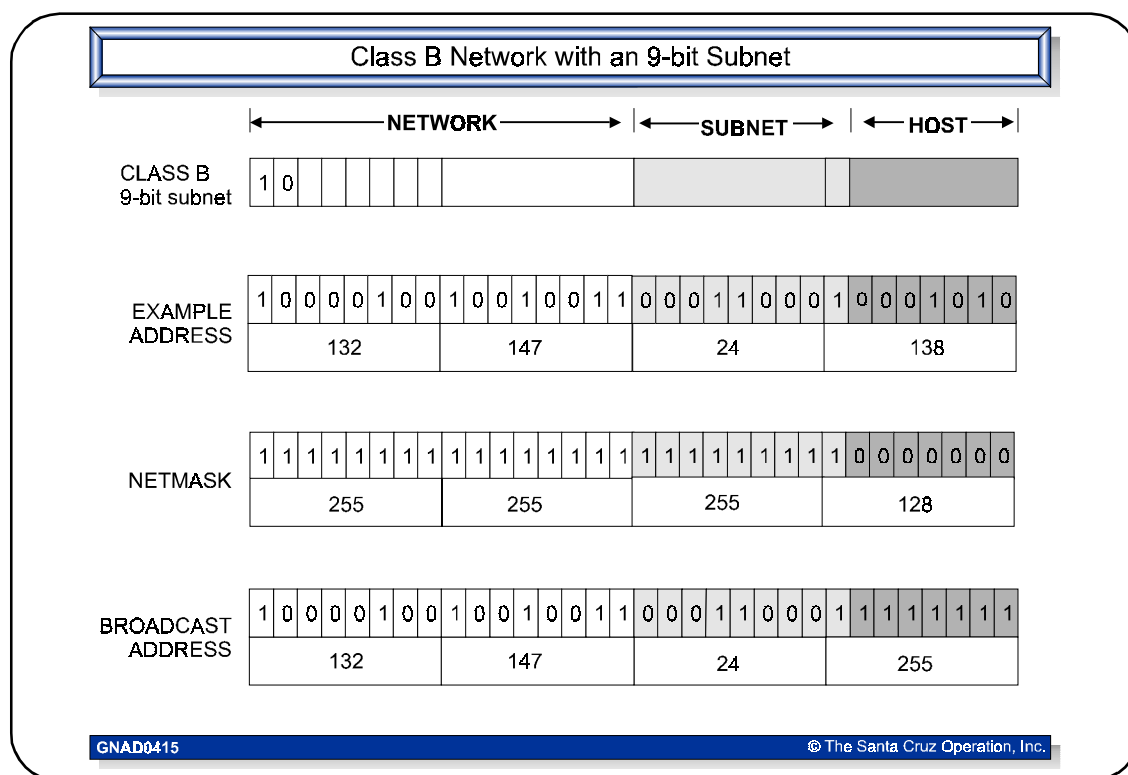
4.2.3.2 Class B non 8-bit subnets

The example in the previous section used a Class B 8-bit subnetting scheme allowing SCO to have up to 254 subnets, each with up to 254 hosts. Although this scheme presently fulfills SCO's needs, it is somewhat wasteful of IP addresses because most of the subnets only have a couple dozen hosts. Furthermore, there will be problems if additional subnets are needed in the future as the company continues to grow.

Because most of the subnets have a small number of hosts, it is not necessary to reserve 8 bits for the host address. A fewer number of bits would suffice, leaving the surplus available for the subnet address.

With 9 bits available for the subnet portion of the address, there are 510 ($2^9 - 2$) possible subnets, each having up to 126 ($2^7 - 2$) hosts.

AV 4-15

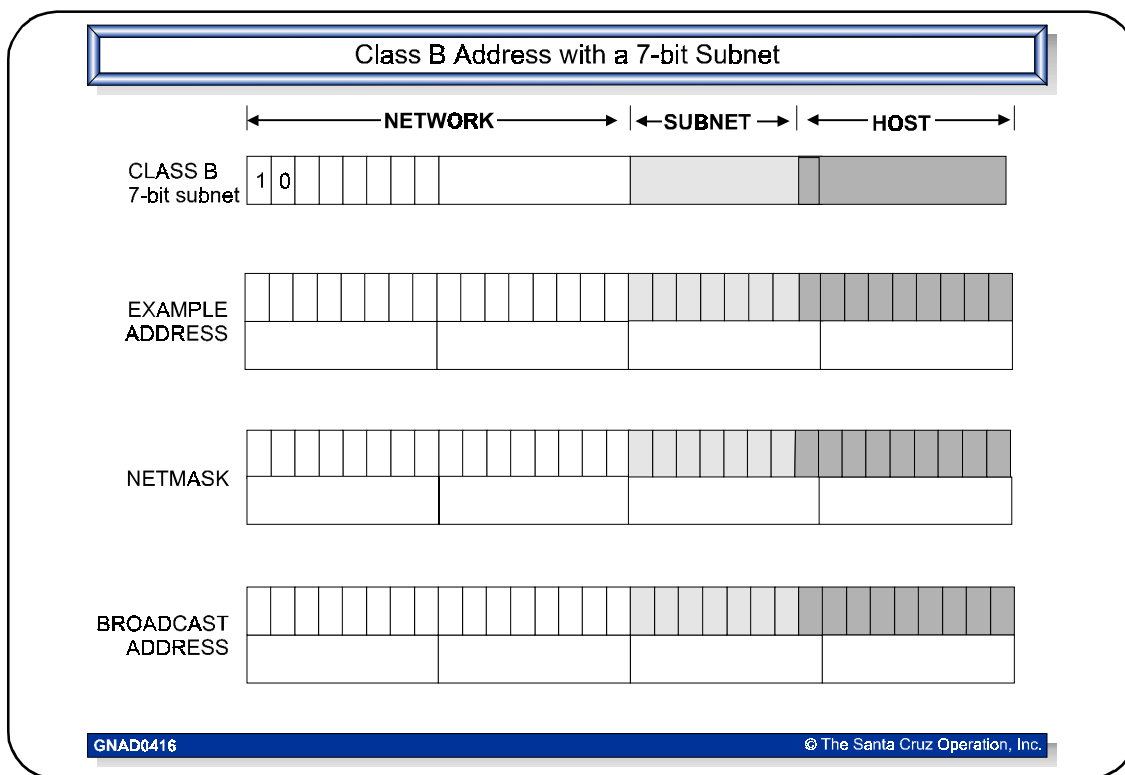


Using the procedure from earlier sections, you can determine the netmask and broadcast values for the IP address 132.147.24.138. Note that this is the same IP address as before, but it is interpreted differently with a 9-bit subnet:

- With an 8-bit subnet, it is host 138 on subnet 24
- With a 9-bit subnet, it is host 10 on subnet 49

- Step 1. Enter the IP address in decimal notation: 132.147.24.138.
- Step 2. Convert the IP address to binary:
10000100 10010011 00011000 10001010
- Step 3. Enter the netmask in binary—enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
11111111 11111111 11111111 10000000
- Step 4. Convert each octet of the netmask to decimal:
255.255.255.128
- Step 5. Enter the broadcast address in binary—enter a one in each host address bit; next copy each network and subnet bit into the corresponding broadcast address fields:
10000100 10010011 0011000 11111111
- Step 6. Convert each octet of the broadcast address to decimal:
132.147.24.255

AV 4-16



Exercise: Creating Class B 7-bit subnets

Use the form in AV 4-16 on page 4-28 to determine the netmask and broadcast address for IP address 175.200.21.8, assuming a 7-bit subnet. What is the subnet number and host number for this IP address, in decimal?

4.2.4 Class C subnets

Class C addresses use three of the available four octets for the network address, leaving only one octet for the host portion of the address. Without subnetting, a Class C network has a limit of 254 nodes and they must all be on the same physical network segment.

To create subnets, a portion of the 8-bit host address must be allocated to the subnet address. Two to six bits can be used for the subnet:

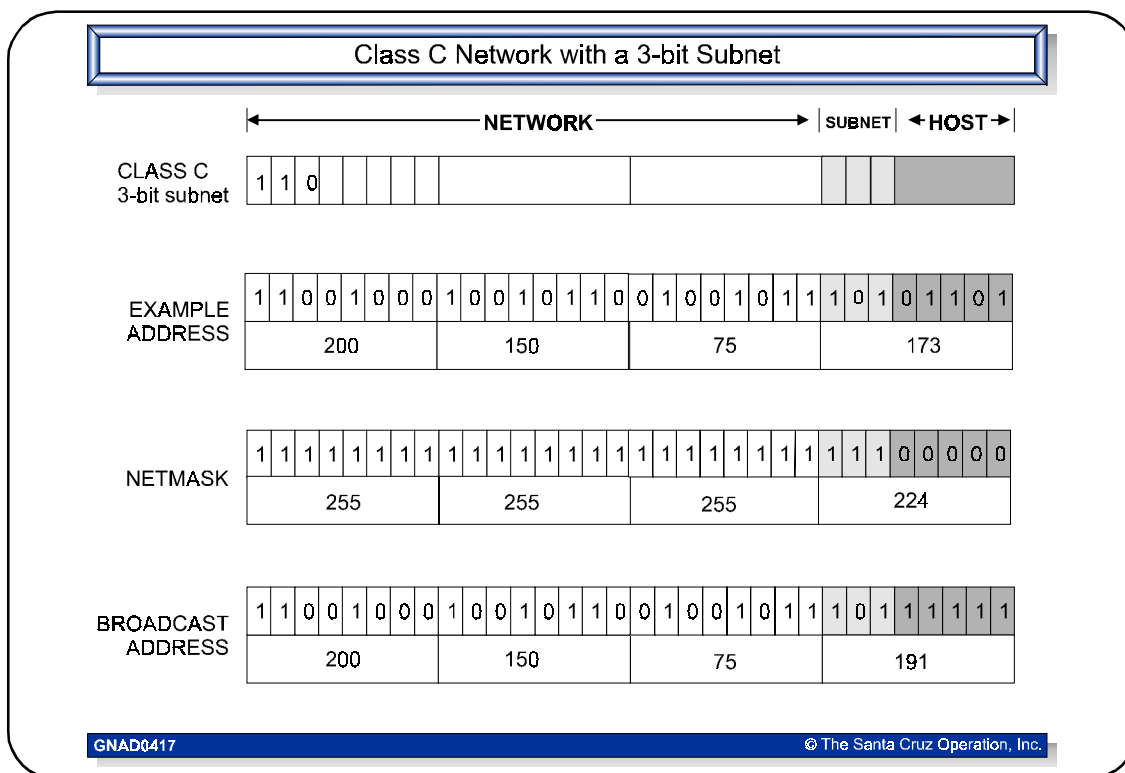
2-bit subnet	2 subnets ($2^2 - 2$); 62 hosts each ($2^6 - 2$)
3-bit subnet	6 subnets ($2^3 - 2$); 30 hosts each ($2^5 - 2$)
4-bit subnet	14 subnets ($2^4 - 2$); 14 hosts each ($2^4 - 2$)
5-bit subnet	30 subnets ($2^5 - 2$); 6 hosts each ($2^3 - 2$)
6-bit subnet	62 subnets ($2^6 - 2$); 2 hosts each ($2^2 - 2$)

The next example shows how the Widget Company uses its Class C network, 200.150.75.0.

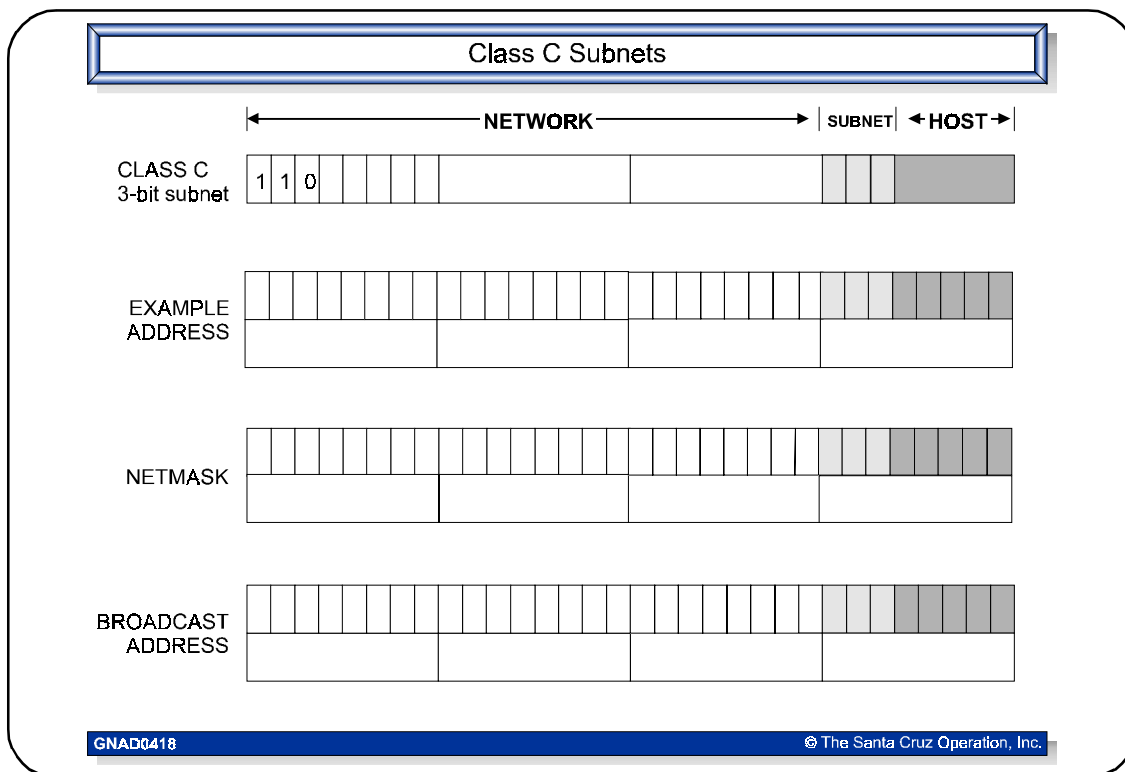
The Widget Company has east, west, north, and south offices that cannot be connected to the same physical network, so it has chosen to implement a 3-bit subnetting scheme. This provides 6 subnets with these network addresses:

subnet 1 (binary 001)	200.150.75.32
subnet 2 (binary 010)	200.150.75.64
subnet 3 (binary 011)	200.150.75.96
subnet 4 (binary 100)	200.150.75.128
subnet 5 (binary 101)	200.150.75.160
subnet 6 (binary 110)	200.150.75.192

AV 4-17



AV 4-18



With the same procedure used in earlier sections, you can determine the netmask and broadcast values for IP address 200.150.75.173:

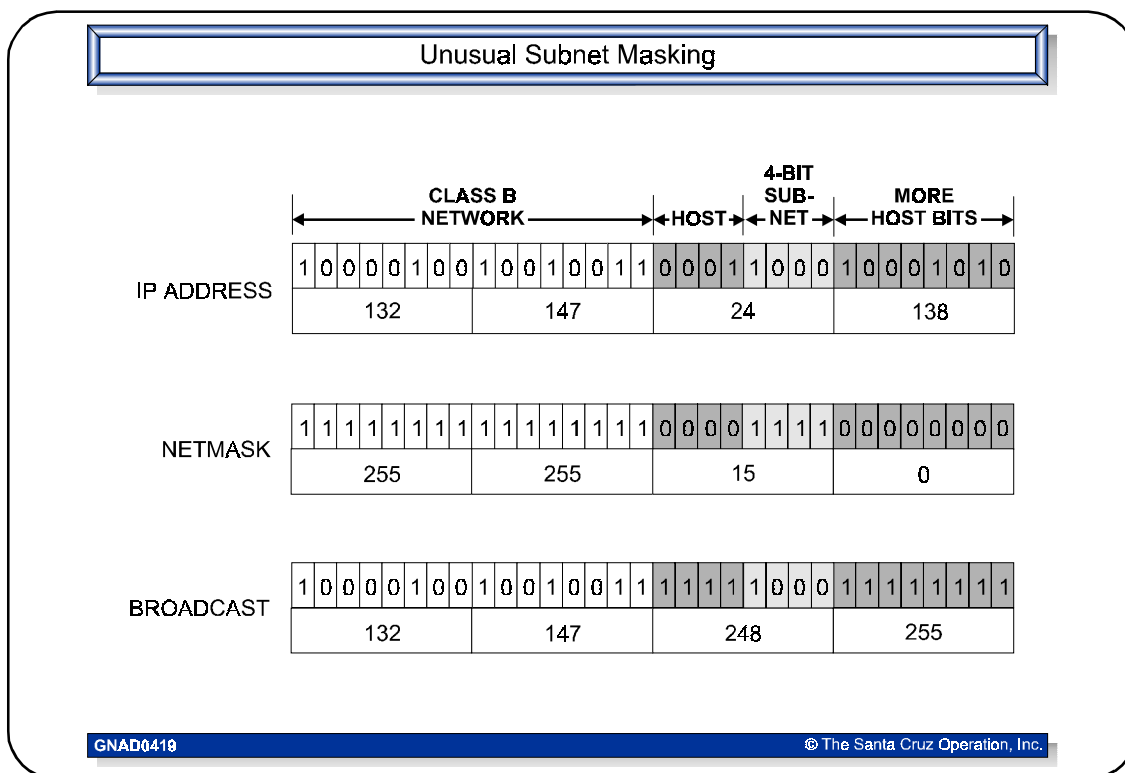
- Step 1. Enter the IP address in decimal notation: 200.150.75.173.
- Step 2. Convert the IP address to binary:
11001000 10010110 01001011 10101101
- Step 3. Enter the netmask in binary— enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
11111111 11111111 11111111 11100000
- Step 4. Convert each octet of the netmask to decimal:
255.255.255.224
- Step 5. Enter the broadcast address in binary—enter a one in each host address bit; next copy each network and subnet bit into the corresponding broadcast address fields:
11001000 10010110 01001011 10111111
- Step 6. Convert each octet of the broadcast address to decimal:
200.150.75.191

Exercise: Class C subnet addresses

Use the form in AV 4-18 on page 4-32 and the Class C IP address in the example above, 200.150.75.173, to give the host and subnet numbers:

- With a 3-bit subnet
- With a 2-bit subnet
- With a 4-bit subnet

AV 4-19



4.2.5 Unusual subnet masks

In all of the examples so far involving subnets, the network and subnet bits have been contiguous. This is because it is easier to convert and otherwise operate on IP addresses when the network portion and the subnet portion are contiguous. This is not a requirement, just a convenience.

You might inherit a situation where the previous network administrator did not use contiguous bits. This results in extra work, but the procedure provided in this lesson still works.

AV 4-19 on page 4-36 illustrates:

- A class B address of 132.147.24.138.
- A 4-bit subnet that is non-contiguous with the network portion, forcing the host portion to also be non-contiguous.

With the netmask applied, this address specifies host 394 on subnet 8.

4.3 Lesson Summary

This lesson described how subnetworks are used to manage large or geographically distant networks. The most common reasons to use subnets are increased network traffic, addition of network nodes, and the requirement to join separate LANs. Critical factors in determining how to organize the network are the number and size of subnetworks needed to accommodate current needs and future expansion.

Subnets are implemented by planning a network topology, configuring each node with an appropriate IP address, netmask, and broadcast address, updating the hostname resolution service, and configuring routers between subnets.

Typical examples of subnets on Class A, B, and C networks are provided, as well as unusual subnet implementations, to help you cope with the many types of subnets you might encounter.

Review questions

1. Describe three organizational growth conditions that might cause you to divide your network into two or more subnets.
2. Without subnetting, how many hosts can exist on a Class C network? If a 4-bit subnet scheme is introduced to this Class C network, how many subnets can there be? How many hosts can exist on each subnet? What is the total number of hosts for this Class C network?
3. Your organization has a Class C network address. Because of geographical considerations, you want to set up 15 subnets. What is the minimum number of subnet bits that will allow you to have these 15 subnets? What is the maximum number of hosts that can exist on each subnet?

4. Which netmask would you recommend for a Class B network that requires 10 subnet bits?

5. A host with IP address 186.35.22.11 has a netmask value of 255.255.240.0. What broadcast address should it use?
Hint: 22=00010110 (binary); 240=11110000 (binary).

6. Which of the following are valid netmask values for a host with IP address 173.199.1.32? Why?

255.255.255.0
255.255.254.0
255.255.255.128
255.255.255.224

7. A node with the IP address 200.220.77.33 has a netmask value of 255.255.255.240. How many bits of the address make up the network, subnet, and host portions?

4.4 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

bc(1)	Invokes a command-line calculator
--------------	-----------------------------------

Relevant files

<i>/etc/hosts</i>	Local table for hostname to IP address matching. See <i>hosts(4tcp)</i> .
-------------------	---

4.5 Answers to Review Questions

1. Three organizational growth conditions that might cause you to divide your network into two or more subnets include: An increase in the number of hosts; several independent LANs need to be combined into a single enterprise LAN; and Increased traffic as more services are more widely dispersed.
2. Without subnetting, 254 hosts can exist on a Class C network. A 4-bit subnet scheme allows for 14 subnets, each of which can have up to 14 hosts. The total number of hosts for this subnetted Class C network is 196 ($14 * 14$).
3. You will need at least 5 subnet bits to support 15 subnets. (Although the number 15 can be represented with 4 bits, it is invalid to set them to all zeros or all ones.) Each 5-bit subnet can have up to 6 hosts.
4. The recommended netmask for a Class B network that requires 10 subnet bits is 255.255.255.192. Other netmasks also specify 10 subnet bits, but they would not form contiguous fields within the address (e.g., 255.255.255.3).
5. A host with the IP address 186.35.22.11 and the netmask value 255.255.240.0 should use the broadcast address 186.35.31.255.
6. A host with the IP address 173.199.1.32 is part of a Class B network.

It can have a netmask value of 255.255.255.0. The IP address 173.199.1.32 specifies Host #32 on Subnet #1.

It cannot be 255.255.254.0 because the IP address 173.199.1.32 would specify Subnet #0, an invalid value.

It can have the netmask value of 255.255.255.128. The IP address 173.199.1.32 specifies Host #32 on Subnet #2.

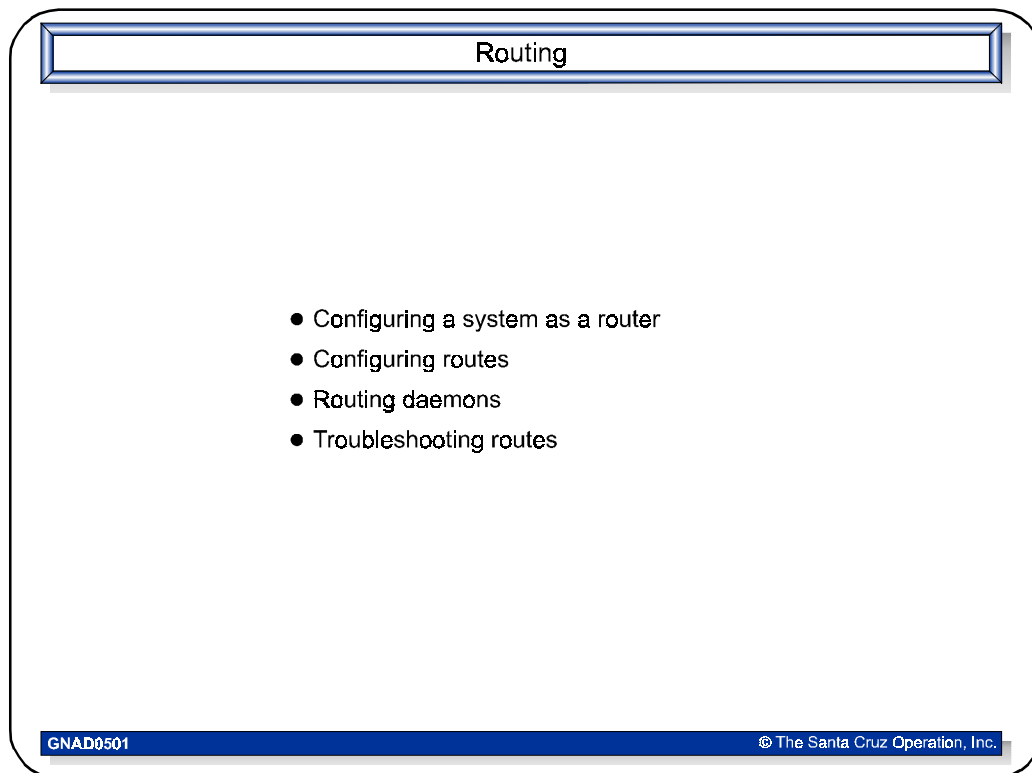
It cannot be 255.255.255.224 because the IP address 173.199.1.32 would specify Host #0, an invalid value.

7. A host with the Class C IP address 200.220.77.33 and a netmask value of 255.255.255.240 uses 24 bits for the network portion, 4 bits for the subnet portion, and 4 bits for the host portion.

5

Routing

AV 5-1



Lesson 5

Routing

Introduction

A router connects two networks together. Networks cannot be connected to control network traffic or extend the network across different media types. Connecting networks also enables network growth when physical configuration limits, such as cable length and number of hosts, are reached.

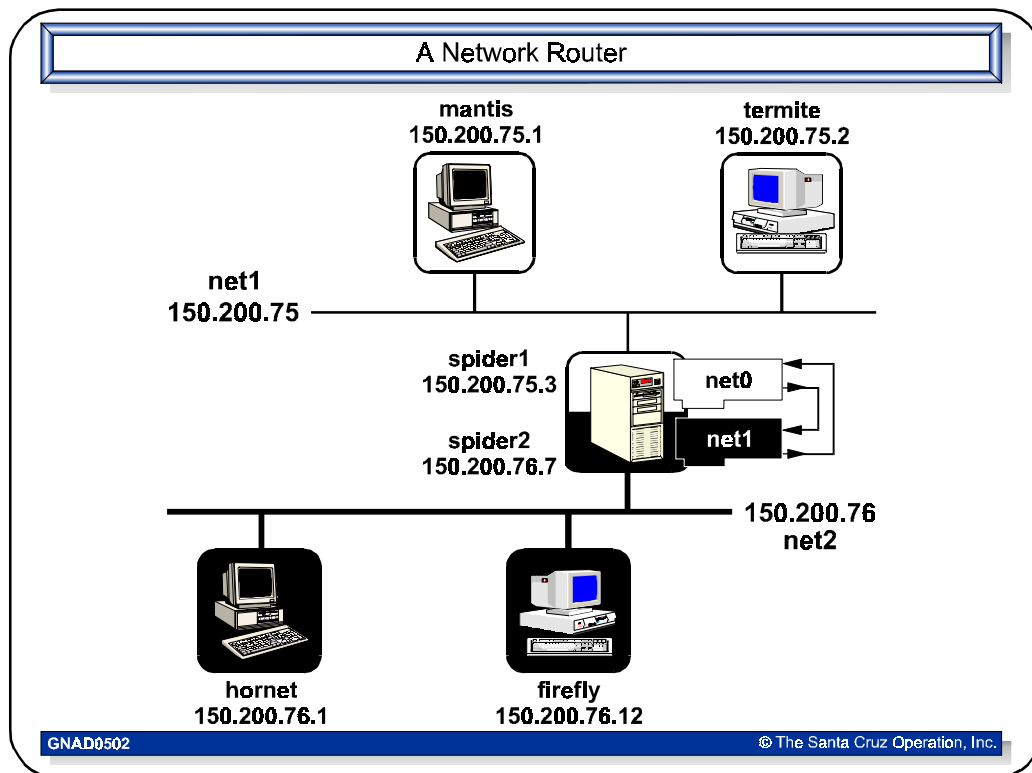
To set up a router, you will add additional network adapters and configure the software. This lesson describes how to do this.

Objectives

This lesson discusses how to set up a system as a router. By the end of this lesson, you will be able to:

- Use the **Network Configuration Manager** to add a second network adapter to a router system
- Configure a routing daemon to enable routing to take place on your system
- Use administrative utilities to track your system's routing capabilities

AV 5-2



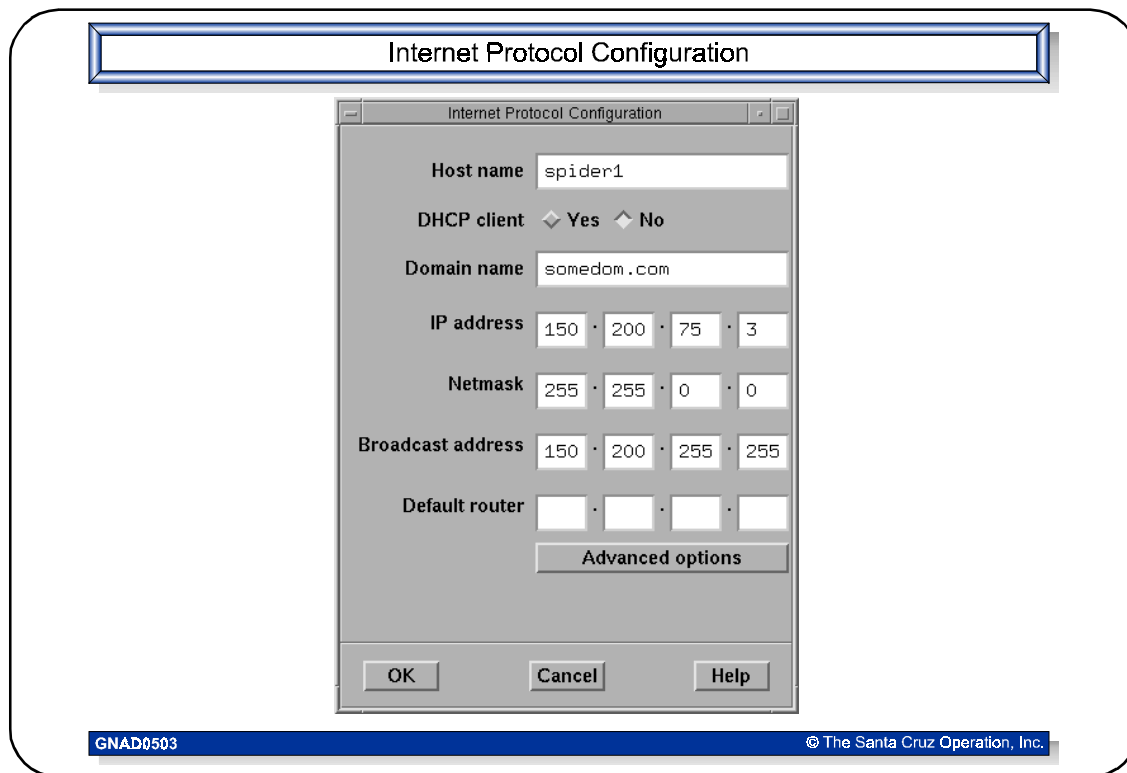
5.1 Configuring a System as a Router

Configuring your system as a router requires:

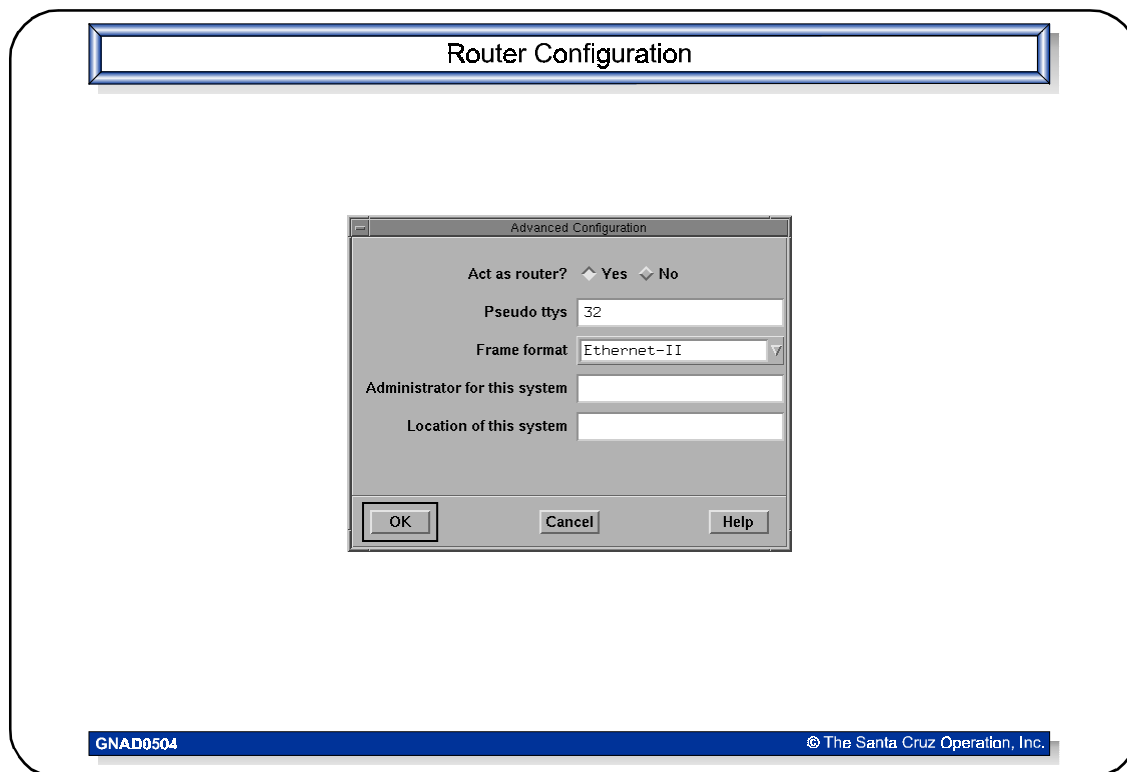
- Installing additional network adapters
- Running the **Network Configuration Manager** or **netcfg(1M)** to configure the software

After configuration, routing between your network and others happens automatically.

AV 5-3



AV 5-4



5.1.1 Installing additional network adapters

These steps describe how to configure a second network adapter:

Step 1. Start the **Network Configuration Manager**.

Step 2. Select **Hardware** ⇨ **Add New Lan Adapter**.

During the configuration, in addition to configuring the network adapters, you also need to configure your system as a gateway.

Step 3. Select **TCP/IP** under any of the LAN adapters currently configured.

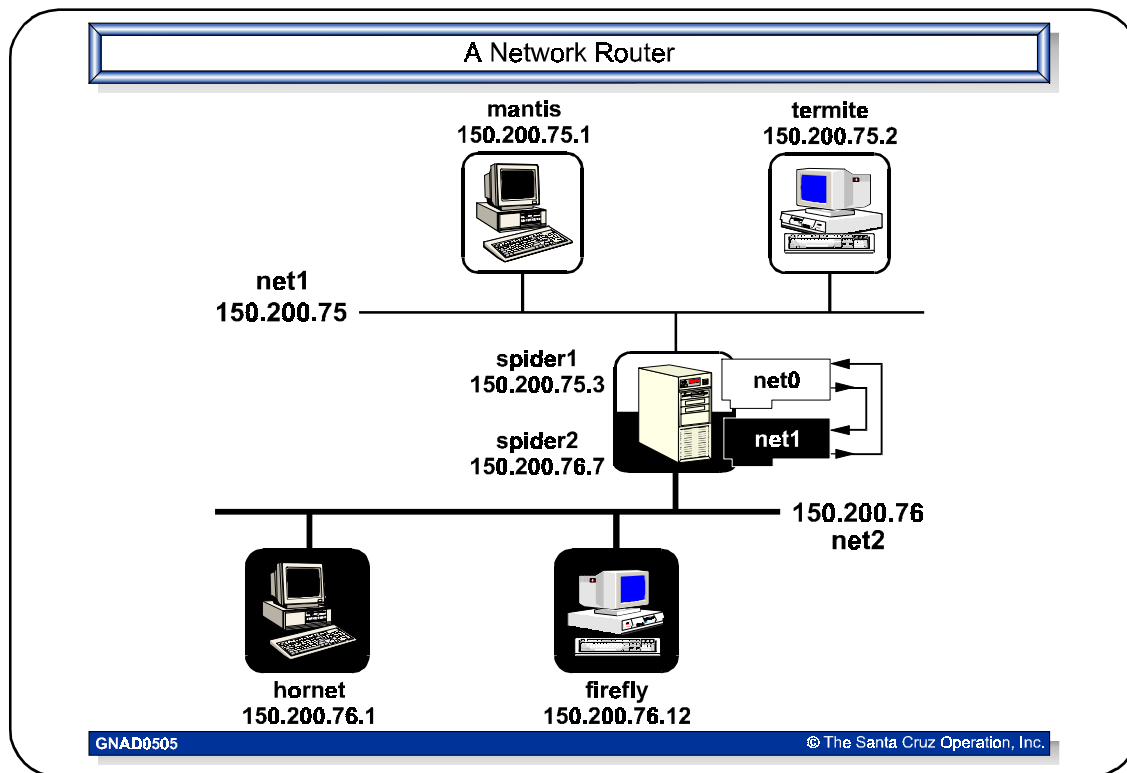
Step 4. Select **Protocol** ⇨ **Modify protocol configuration** from the top menu. The form shown in AV 5-3 displays.

Step 5. Click on **Advanced options** in the **Internet Protocol Configuration** form. The form shown in AV 5-4 displays.

Step 6. Verify that “Act as router?” is set to **Yes**.

Step 7. Click on **OK** in the Advanced Configuration form.

AV 5-5



5.1.2 Kernel parameters

When the **Network Configuration Manager** is used to configure your system as a router, kernel parameters are set. One of these parameters is `ipforwarding`.

When `ipforwarding` is set to one (1), the system routes IP packets between any of its own network interface cards. For example, in AV 5-5, the router has two network interface cards:

- `net0` which connects to network 150.200.75
- `net1` which connects to network 150.200.76

If the router receives a packet from network 150.200.75 with a destination of 150.200.76, the router forwards the packet from `net0` to `net1`.

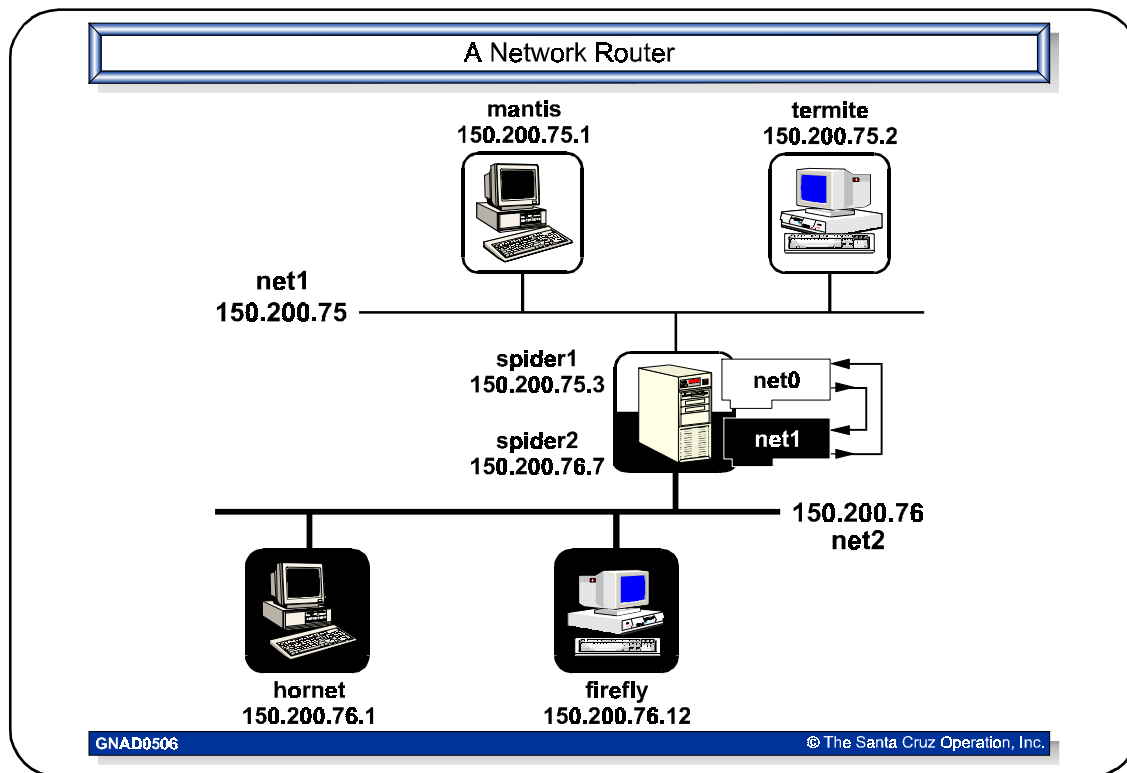
When set to zero (0), the system does not route IP packets between its own network interfaces. This means that systems on one network cannot access systems on any other network without first logging into the router.

Setting `ipforwarding` to 0 allows the network or system administrator to control who can access a network. Only those who have an account on the router are able to access other networks.

Note that after the parameter is set for one adapter, you do not have to set it for the other interfaces on the router. When you indicate to **the Network Configuration Manager** that you are configuring a gateway, it sets `ipforwarding` to one (1); otherwise it sets it to zero (0).

You can also use the `inconfig(1Mtcp)` command to make modifications to this and other kernel parameters. This command modifies the `/etc/inet/inet.dfl` file. **WARNING!** This file should *not* be edited manually!

AV 5-6



5.2 Configuring Routes

The IP routing mechanisms consist of the following:

- A kernel routing table with direct and indirect routes
- Routing protocols
- Routing daemons and configuration files that implement the routing protocols
- Configuration and troubleshooting utilities

5.2.1 Direct and indirect routing

Two types of routing exist:

<i>Direct</i>	A direct route is used when a machine can send an IP packet to another machine without going through a router. For example, in AV 5-6, a direct route is used between <i>hornet</i> and <i>firefly</i> .
<i>Indirect</i>	An indirect route is used when a machine must send the packet through a router to reach the second machine. For example, in AV 5-6, an indirect route is used between <i>hornet</i> and <i>mantis</i> (through the <i>spider1</i> and <i>spider2</i> interfaces on the router).

AV 5-7

Kernel Routing Table on *mantis*

<u>Destination</u>	<u>Netmask</u>	<u>Gateway</u>	<u>Interface</u>
127.0.0.1	255.0.0.0	127.0.0.1	lo0
150.200.75.1	255.255.255.0	127.0.0.1	lo0
150.200.75	255.255.255.0	150.200.75.1	net0
150.200.76	255.255.255.0	150.200.75.3	net0
224	255.0.0.0	150.200.75.1	net0
default	N/A	150.200.75.3	net0

GNAD0507

© The Santa Cruz Operation, Inc.

5.2.2 Routing table

Every system running SCO TCP/IP has a routing table within the kernel. During TCP/IP startup, the routes added to this table include a route:

- To the host 127.0.0.1 (*localhost*) through the *lo0* loopback adapter
- To the IP address of each network adapter. In the previous example for *mantis*, a route is added for the IP address 150.200.75.1 through its *lo0* adapter.
- To the network that each network adapter is on. In the previous example for *mantis*, a route is added for the 150.200.75 network through the *net0* adapter.
- To a multicast network

AV 5-7 shows some of the entries that are in the kernel routing table for *mantis* (in the previous example). The fields in the table include:

- Destination network IP address
- Netmask
- Next hop (gateway), the next machine that knows how to reach the ultimate destination machine
- Identity of the network interface through which the packet is sent to reach the next hop

AV 5-8

Output of `netstat -rn` on *mantis*

Destination	Gateway	Flags	Refs	Use	Interface
default	150.200.75.3	UGS	1	8443	net0
127.0.0.1	127.0.0.1	UH	2	351	lo0
150.200.75/24	150.200.75.1	UC	1	392	net0
150.200.76/24	150.200.75.3	UGS	0	0	net0
150.200.75.1	127.0.0.1	UGHS	4	1097	lo0
224/8	150.200.75.1	UGS	1	0	net0

GNAD0508

© The Santa Cruz Operation, Inc.

5.2.3 The netstat(1Mtcp) command

Use the **netstat**(1Mtcp) command with the **-r** option to view the kernel's routing table. You can include the **-n** option to show numerical IP addresses instead of symbolic names. This can sometimes be helpful in uncovering network problems obscured by the symbolic names.

The **netstat**(1Mtcp) command displays this information:

destination/# The network or host to which this route allows you to connect. In the case of a multicast route, it gives the prefix and mask of the destination.

 The number after the slash indicates the number of network ID bits in the address.

gateway The name of the gateway configured for this route. If directly connected, this is a local address; otherwise, it is the name of the host through which the packets must be routed.

flags The state of the route.

refs The current number of active connections using the route.

use The current number of packets sent using this route.

interface The name of the physical network interface used to begin the route.

Exercise: Examining routes with netstat(1Mtcp)

Enter the **netstat** command with the **-r** option. Try it again with the **-rn** option.

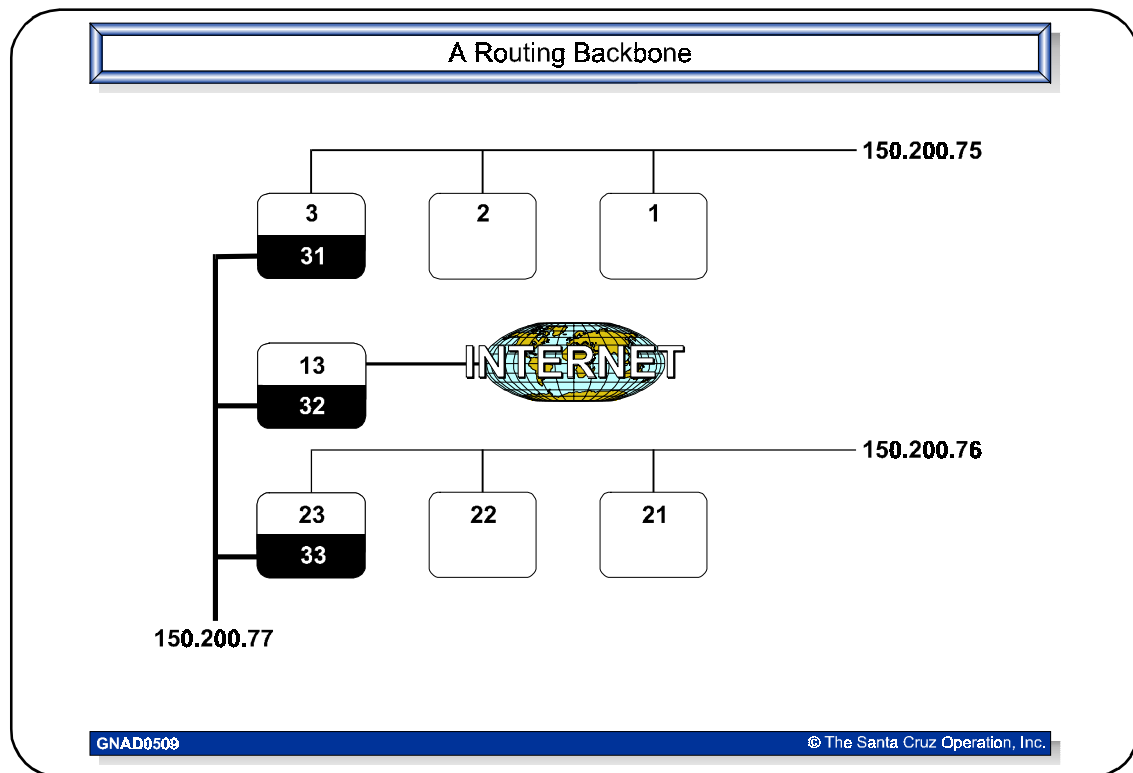
Refer to the **netstat(1Mtcp)** manual page to interpret the output.

5.2.4 Updating routes

Entries in the kernel routing table are created, deleted, and changed:

Automatically	When running the Network Configuration Manager to configure or remove a network interface. Through this process, a system always knows about the networks to which it is directly connected.
Manually	Using the route(1Mtcp) command, a system can be configured to use specified routes for communicating with networks to which it is not directly connected.
Dynamically	Using specialized routing daemons running on the system. These daemons use routing protocols to communicate with routing daemons on other machines. This allows the daemons to learn about the existence of routes to networks other than the networks to which the machine is directly connected.

AV 5-9



5.2.5 Manual updates and route(1Mtcp)

The **route(1Mtcp)** command makes manual updates to the kernel's routing tables. In general, **route** has the form:

```
route [-nqv] keyword keyword-specific-arguments
```

where *keyword* is one of:

add	Adds a route
change	Changes aspects of a route (such as its gateway)
delete	Deletes a specific route
flush	Removes all gateway entries from the routing tables
get	Looks up and displays the route for a destination
monitor	Continuously reports any changes to the routing information base, routing lookup misses, or suspected network partitions

Given the network topology illustrated in AV 5-9, use the **route** command to:

- Add a route to the 150.200.77 network for hosts 1 or 2 through host 3. Enter:

```
route add -net 150.200.77 150.200.75.3
```

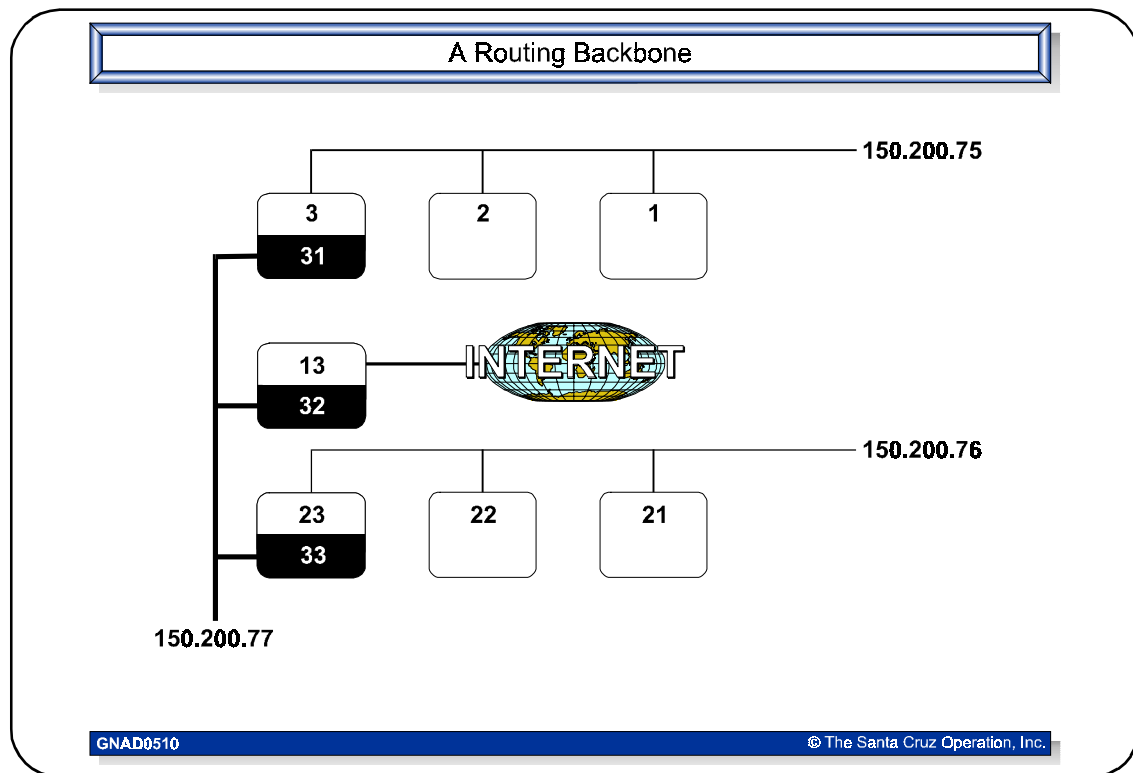
- Add a route to the 150.200.76 network for hosts 1 or 2 through host 3. Enter:

```
route add -net 150.200.76 150.200.75.3
```

While this approach works, it has some notable drawbacks:

- These routes still do not tell the IP software how to reach machines on the Internet (connected through host 13/32).
- All routes for hosts 1 and 2 go through the same host, 3, but are listed separately. This is not efficient. In this case, use a *default route* (see Section 5.2.6, “Default routes,” on page 5-21).

AV 5-10



5.2.6 Default routes

If a given router serves as a route to a much larger network, such as the Internet, or if a host has contact with only one router it might be easier and more efficient to configure the hosts to use a *default route*.

Default routes:

- Are used to tell a system where to send packets if the system does not have a known route to a network
- Allow systems to keep small routing tables by only having entries for networks to which they are directly attached, and a default route
- Are very common in the LAN environment

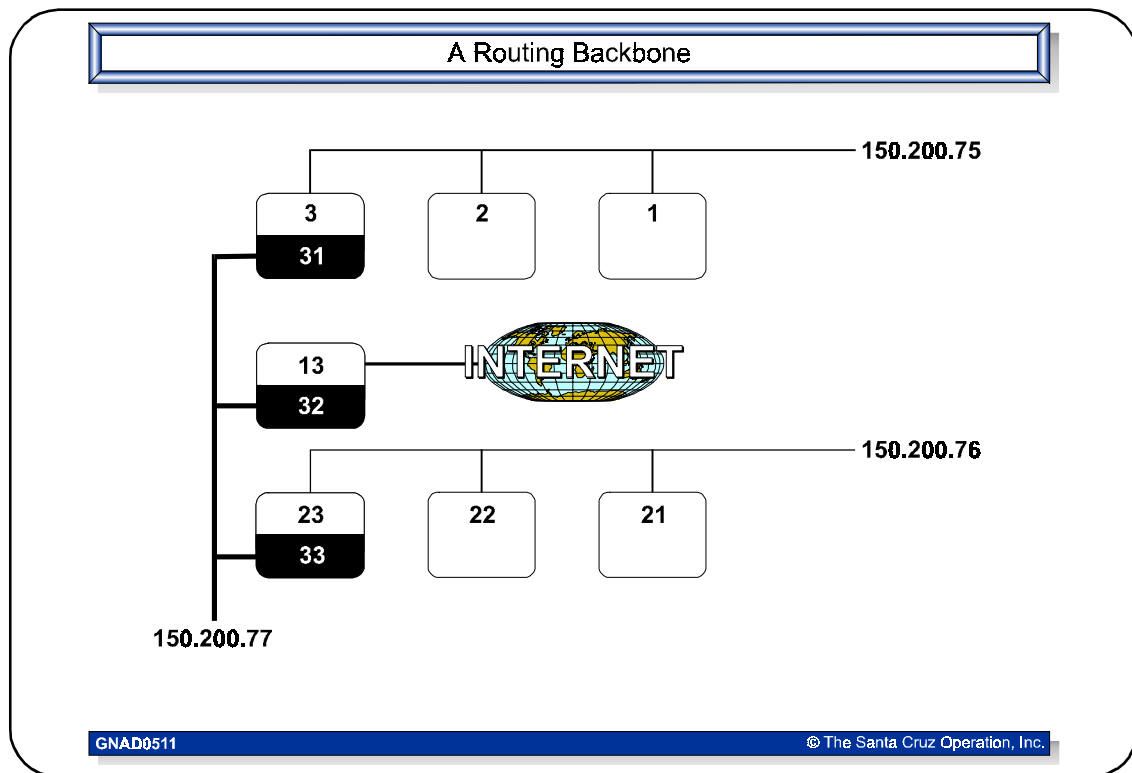
For example, consider AV 5-10:

- Machines 1 and 2 have machine 3 as their default route.
- Machines 21 and 22 have machine 23 as their default route.
- Machines 3/31 and 23/33 have machine 32 as their default route.
- Machine 13/32 has the ISP's router as its default route.

For machines 1, 2, 21, or 22 to send a packet to the Internet using this scenario:

1. The machines forward the packet to their default routers (3 and 23 respectively).
2. The default routers use their default routes to forward the packet to machine 32.
3. Machine 32 forwards the packet to the ISP, which has routes that eventually get the packet to its destination.

AV 5-11



Default routes are configured:

- Temporarily using the **route**(1Mtcp) command. For example, to create a default route on either *host1* or *host2* in AV 5-11, enter:

```
route add default 150.200.75.3
```

- Permanently using:
 - An entry in the */etc/inet/config* file. This is what happens with the default route specified at initial system load (ISL). For example, to create a default route on either *host1* or *host2* using the */etc/inet/config* file, enter this into the file:

```
4c:/usr/sbin/route::y::add default 150.200.75.3
```

- An entry in the */etc/inet/gateways* file if you are running the **routed**(1Mtcp) routing daemon (see Section 5.3.1, “*routed*(1Mtcp),” on page 5-27). For example, to create a default route on either *host1* or *host2* using the */etc/inet/gateways* file, enter this into the file:

```
net 0.0.0.0 gateway 150.200.75.3 metric 1 passive
```

NOTE: When you use **netcfg**(1M) to modify the TCP/IP protocol and set a default route, that route is recorded in this file in a similar way.

See Section 7.2.1, “The *config*(4tcp) file,” on page 7-13 for more information on the */etc/inet/config* file.

5.3 Dynamic Routing Table Updates and Routing Daemons

So far, routing tables have been updated manually, with either:

- Commands entered from the command line
- Entries placed in files

If your network topology changes regularly, or you would like your hosts to determine their routes automatically, you must configure your hosts to run a *routing daemon*. Note that using a routing daemon instead of static or default routes increases network traffic and CPU usage.

Routing daemons update routing tables dynamically. These routing daemons are provided with UnixWare 7:

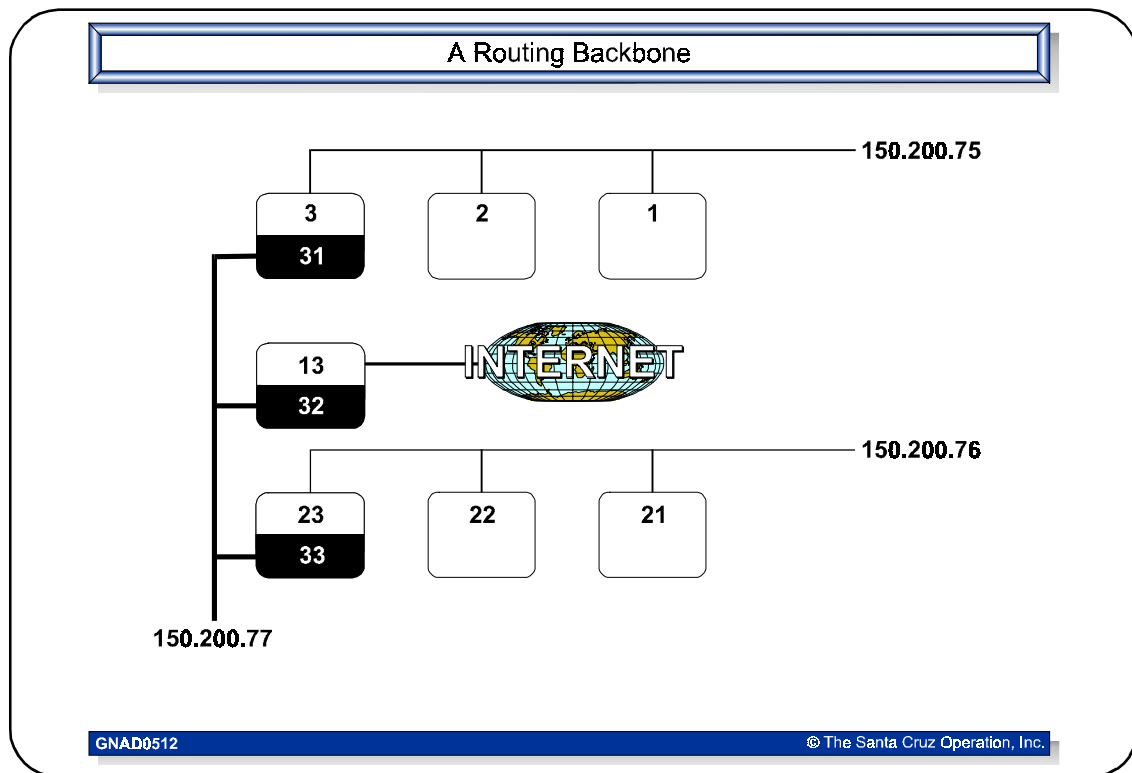
routed(1Mtcp) Is invoked at boot time to manage the network routing tables

gated(1Mtcp) Is the gateway routing daemon

mrouted(1Mtcp) Is the Internet Protocol routing daemon

How these daemons are chosen and a discussion of each is described in the next sections.

AV 5-12



5.3.1 routed(1Mtcp)

The **routed** daemon starts by default on UnixWare 7 hosts (unless you have configured the **gated**(1Mtcp) daemon). It is usually not necessary to do any configuration for **routed**.

Features of **routed** include:

- Dynamic updates to the kernel's routing table

Updates are based on information that **routed** learns from other routers on networks to which the machine running **routed** is directly attached. For example, in AV 5-12, routing daemons on hosts 3/31, 13/32, and 23/33 can all exchange routing information.

Remember, hosts 1, 2, 21, and 22 do not need to run **routed**. They can use a default route to hosts 3 and 23 respectively.

- Routing protocols implemented:
 - Routing Information Protocol (RIP) versions 1 and 2
 - Internet Router Discovery Protocol (IRDP)

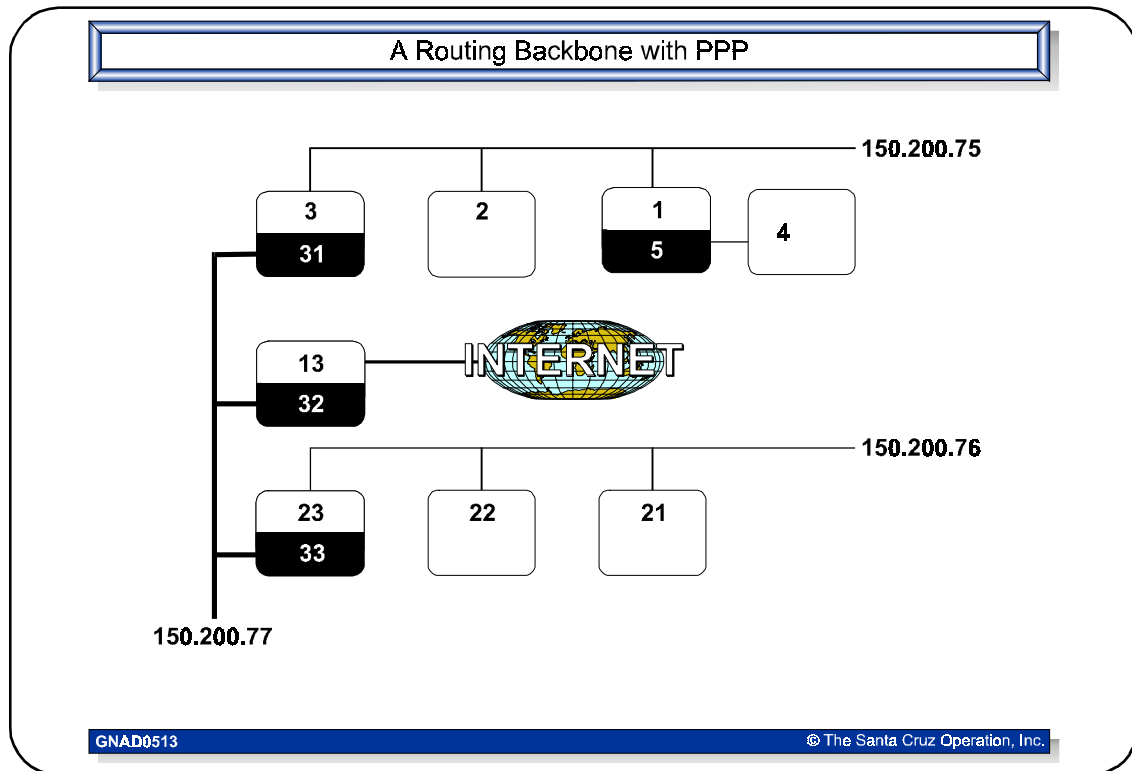
The **routed**(1Mtcp) daemon:

- Is started by default from an entry in the */etc/inet/config* file, but only if **gated**(1Mtcp) is not able to start. For further information, see Section 7.2.1, “The config(4tcp) file,” on page 7-13.
- Has several command-line options that control its behavior. Refer to **routed**(1Mtcp) for additional information.

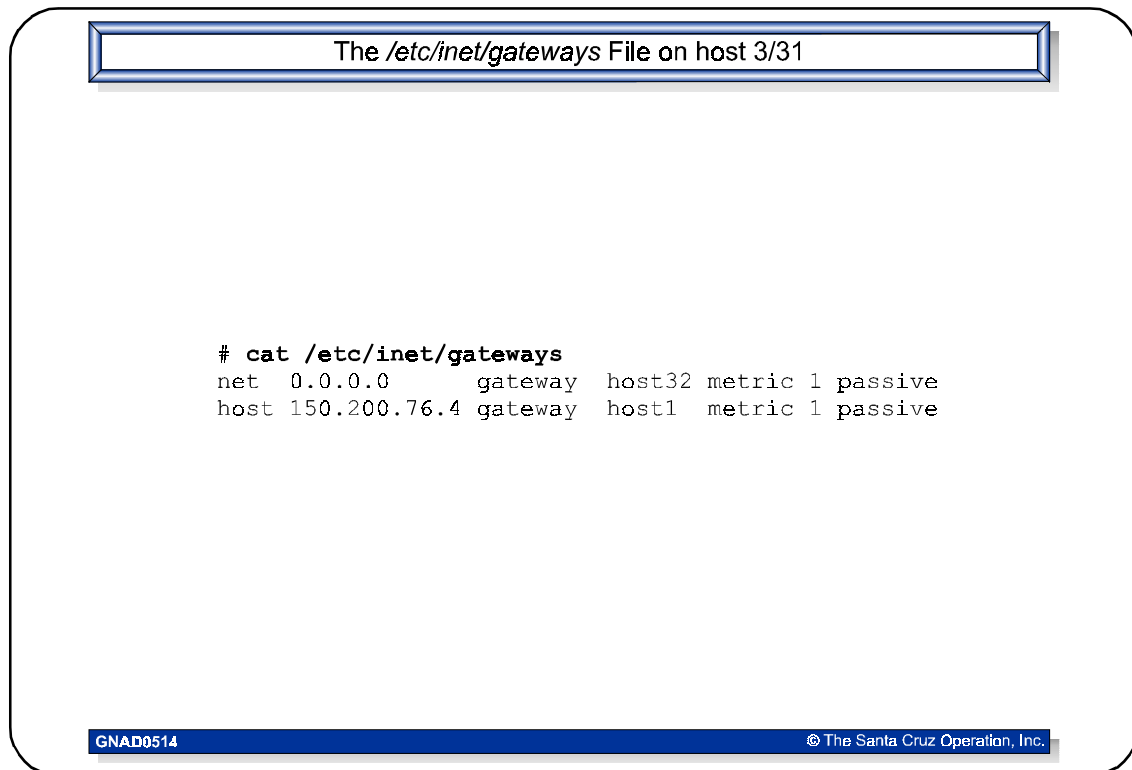
When **routed** starts, it:

1. Reads the */etc/inet/gateways* file, if present, and installs the routes defined there into the kernel’s routing table.
2. Broadcasts on each local network to advertise its routes to any other nodes directly attached to its networks.
3. If other routers exist they continually cooperate in maintaining a globally consistent view of routing by exchanging routing data using one of the routing protocols mentioned above.

AV 5-13



AV 5-14



Normally, this startup procedure and the subsequent protocol exchanges by routers find and keep routes up to date. You only create the */etc/inet/gateways* file and place routes there if:

- Your router is on an ATM or similar media that does not support broadcasting. Such routers cannot be discovered through broadcast exchanges of protocol information, the default method for **routed**.
- Routers on the network are running incompatible routing protocols. For example, in AV 5-13, if host 13/32 was running a protocol other than RIP or IRDP, then host 3/31 and 23/33 would not be able to discover the route to the Internet.
- There are dynamic routes which will not be discovered through the procedures described so far. For example, consider host 4 in AV 5-13:
 - The link between host 4 and 1/5 is through a point-to-point protocol. For example, PPP or SLIP.
 - Host 1/5 is not running a routing daemon.

The route to host 4 is not known to host 3/31, or to the rest of the world.

The */etc/inet/gateways* entries in AV 5-14 solves these last two problems by providing a default route through host 32 and a route to host 4 through host 1.

Each entry in the */etc/inet/gateways* file has the following format:

```
net|host name1 gateway name2 metric val active|passive|external
```

where:

net or host	Indicates if the route is to a network or to a specific host.
name 1	Is the symbolic name of the destination network or host (from <i>/etc/hosts</i>), or the dotted-decimal IP address. This field can also contain the special address 0.0.0.0 to indicate that a default route is being created.
name2	Is the name or address of the gateway to which messages should be forwarded.
val	Is the hop count to the destination host or network.

The final parameter is one of these values:

active	The gateway is expected to exchange routing information. If the router does not exchange routing information, it is dropped from the kernel's routing table.
passive	The gateway is not expected to exchange routing information. The route is initially placed in the kernel's routing table and then left there (unless manually removed). Passive routes are not propagated to other routers using RIP exchanges.
external	The gateway does not communicate directly. The route is not yet entered into the kernel's routing table. If a RIP or IRDP packet proposes the route, it is installed. No other route to this destination is allowed to be installed.

5.3.2 **gated(1Mtcp)**

You might have a network topology that requires a more sophisticated approach than default routes or the routing protocols used by **routed(1Mtcp)**. These cases include:

- You are using your UnixWare 7 system as a router to an autonomous system (AS).
- Your network topology has loops such that hosts can choose between routers, and you want the hosts to do load balancing between the routers.

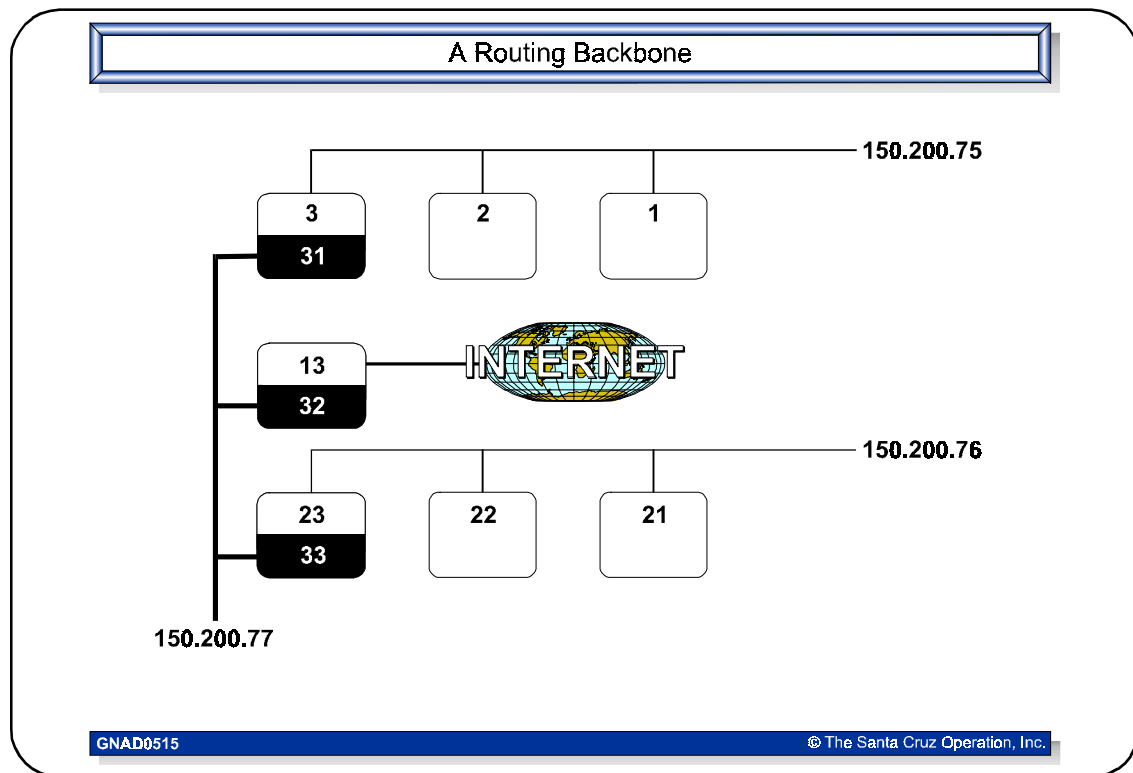
In these cases, you must use the **gated(1Mtcp)** routing daemon to run one or more of these routing protocols:

- Exterior Gateway Protocol (EGP). This protocol is used on the *standard* Internet to route packets between autonomous systems.
- Border Gateway Protocol (BGP). This protocol is used on the NSF-net portion of the Internet to route packets between autonomous systems.
- Open Shortest Path First (OSPF). This protocol is used on a LAN. Its features include:
 - Dynamic load balancing — If multiple routes exist to a given host or network, each route having the same cost, OSPF splits the load over the available routes evenly. None of **routed**'s protocols do this. This can be very important on large intranets.
 - Type of service routing — Network administrators can specify that different kinds of IP packets should take different routes. For example, voice and video packets might be routed through the 100Mbps connection and other data might be routed through a 10Mbps connection.
 - Scalability — For the above and other reasons, OSPF scales better than the protocols supported by **routed**.

You can also configure **gated(1Mtcp)** to use RIP, RIP2, and IRDP just as **routed(1Mtcp)** does.

The **gated(1Mtcp)** manual page discusses how to configure **gated** to run these protocols. Further discussion of **gated(1Mtcp)** is beyond the scope of this course.

AV 5-15



5.3.3 mrouted(1Mtcp)

Recall from Section 2.2.6, “Multicast addresses,” on page 2-21 that multicast addresses allow data to be sent from a server program to multiple client programs, even those separated by routers. For example, Real Audio® can use multicasting to transmit data to multiple Real Audio players at the same time.

Since modern routing daemons support multicast-aware routing protocols (like IRDP), you probably do not have to do anything to use multicasting on your intranet.

You only have to configure the **mrouted(1Mtcp)** daemon if both of these statements are true:

- You are running multicast programs that need to communicate on different networks or subnets.
- One or more of the routers between the server and client programs are not forwarding multicast packets.

If both of the above are true, you must create a multicast tunnel through the unicast routers by:

1. Configuring **mrouted** to run on a host on the network reachable by the multicast server.
2. Configuring **mrouted** to run on a host on the network reachable by the multicast clients.

The **mrouted** daemon is configured by entries in the */etc/inet/mrouted.conf* file.

For example, in AV 5-15,:

- Host 1 is running a multicast server.
- Host 21 is running a multicast client.
- Host 23/33 is not forwarding multicast packets.

In this case, place the following entries in the */etc/inet/mrouted.conf* file to allow hosts 1 and 21 to exchange multicast packets:

- On host 1, add the entry:

```
tunnel 150.200.75.1 150.200.76.21
```

- On host 21, add the entry:

```
tunnel 150.200.76.21 150.200.75.1
```

Now, each of the machines will re-multicast packets received from the other on their own segments. If those segments contained other machines with routing daemons that are capable of forwarding multicast packets, those routers will forward the multicast packets they received.

NOTE:

- The */etc/inet/mrouted.conf* file exists in default installations. However, because there are no multicast routes configured in this file, **mrouted** terminates shortly after starting.
- You can run **mrouted** with either **routed** or **gated**.

Refer to **mrouted(1Mtcp)** for more information about **mrouted** and the *mrouted.conf* file.

5.4 Troubleshooting Routes

In addition to the **netstat**(1Mtcp) command (discussed in Section 5.2.3 on page 5-15), there are several commands and files that you can use to troubleshoot routing related problems. These include:

- **ping**(1Mtcp)
- **tracert**(1Mtcp)
- **rtquery**(1Mtcp)
- **ripquery**(1Mtcp)

5.4.1 The **ping**(1Mtcp) command

Recall from Lesson 3: “Configuring a Network Adapter,” that you can use the **ping**(1Mtcp) command to verify connectivity of the hosts on your network. (See Section 3.2.2 on page 3-14.)

You can also use **ping** to verify that routes are working between two hosts. Verify that routes are working between hosts by using **ping** to check connectivity. If **ping** does not work, follow these steps:

(Note the first three steps were discussed in Section 3.2, “Testing TCP/IP Connectivity,” on page 3-12).

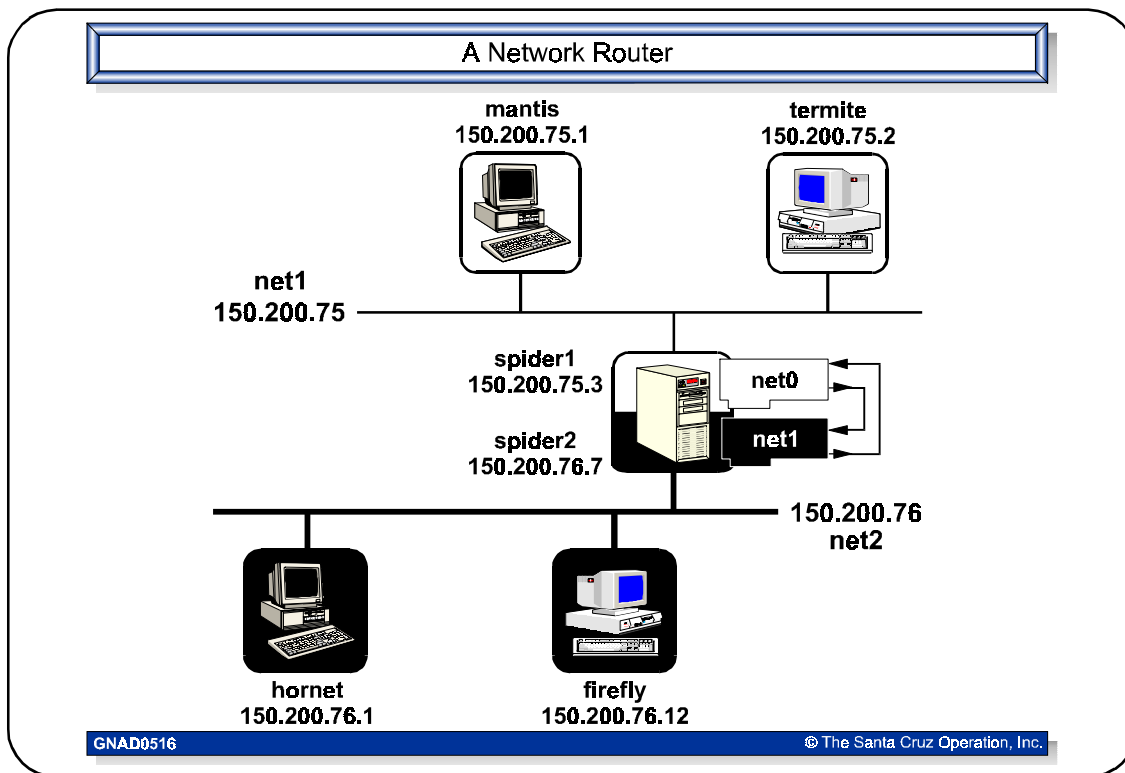
Step 1. Verify the local protocol stack is working. Enter:

ping localhost

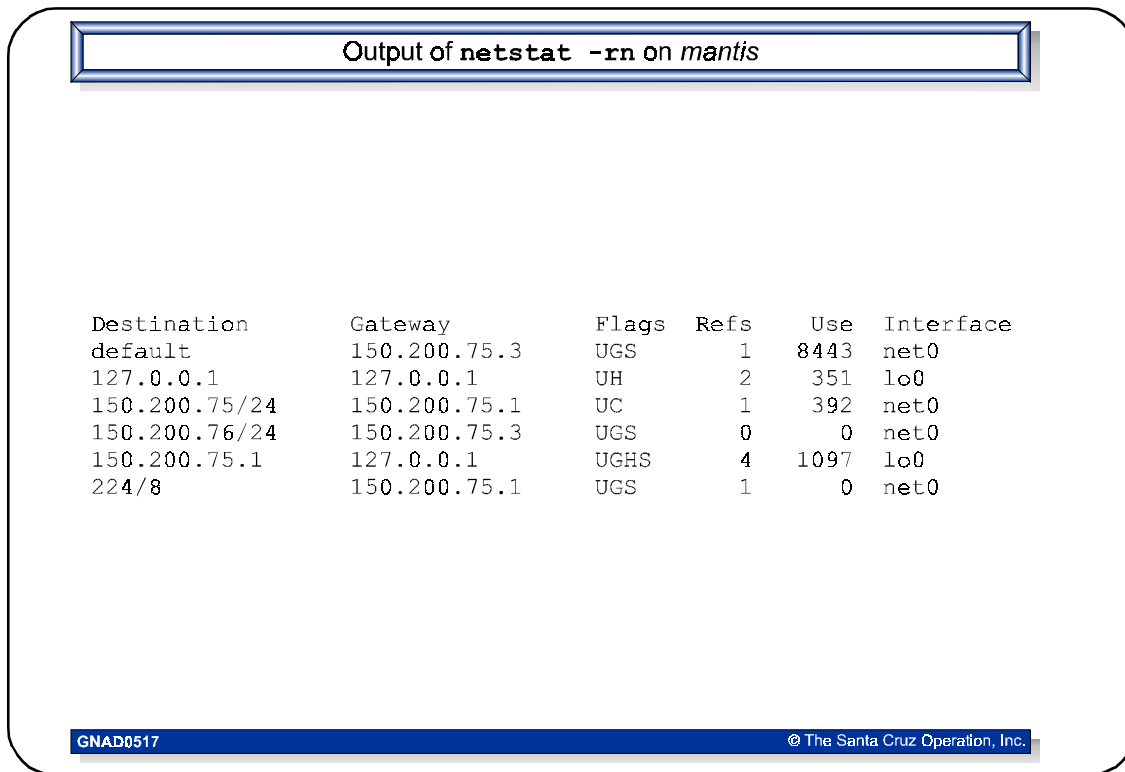
Step 2. Verify the network adapter is working by using **ping** with the broadcast of the network adapter.

Step 3. Verify the router is receiving our packets by using **ping** with the IP address of the router that shares the same network address as the host on which you are running the **ping** command.

AV 5-16



AV 5-17



Step 4. Verify the router is forwarding packets by using **ping** with the IP address of the router that does not have the same network address as the host from which you are running the **ping** command. If this does not work, check:

- The IP protocol stack on the router
- That the *ipforwarding* parameter is set to **1** on the router (see Section 5.1.2, “Kernel parameters,” on page 5-9)

If you have not yet isolated the problem, use the **tracert**(1Mtcp) command.

5.4.2 The **tracert**(1Mtcp) command

Use the **tracert**(1Mtcp) command to determine whether a route is working, and if it is not, where it is broken. The command:

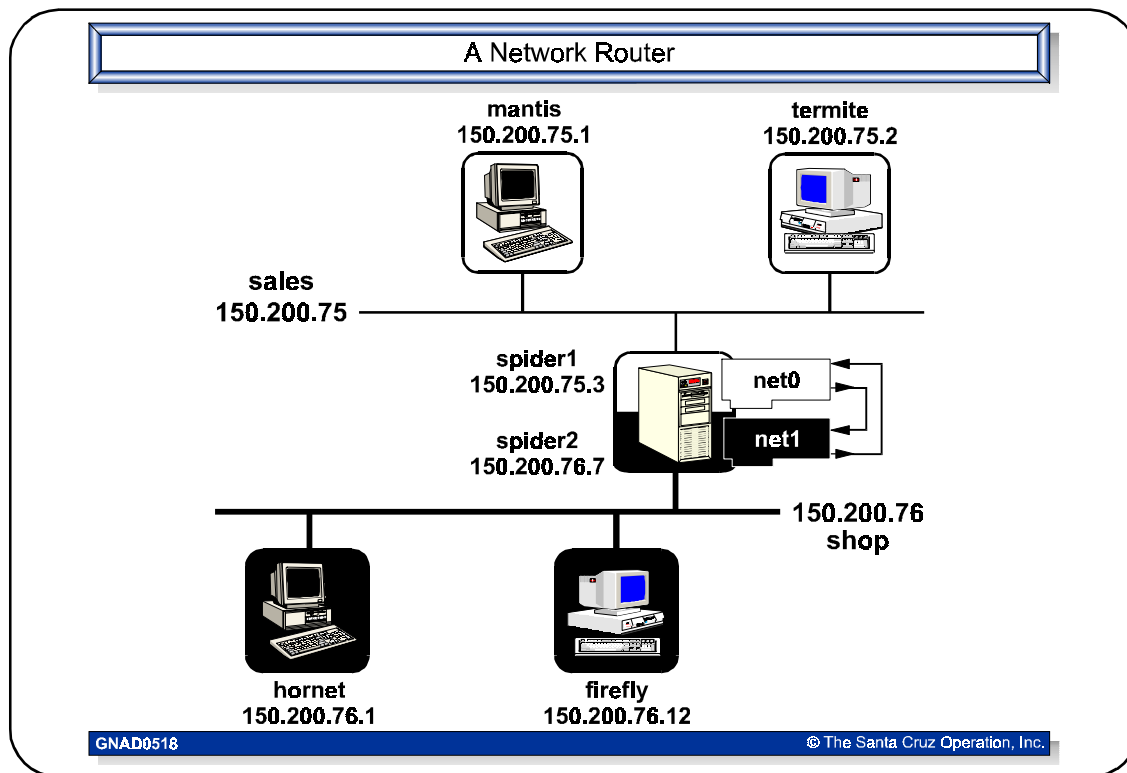
1. Attempts to trace the route that an IP packet follows to a specified host by launching UDP probe packets that have a small *ttl* (time-to-live) value.
2. Listens for an ICMP **TIME EXCEEDED** reply from a router. Several probes are sent with different *ttl* values.
3. If the probe answers come from different routers, the address of each responding router is displayed. If there is no response within 3 seconds, an asterisk (*) is printed for that probe.

The basic syntax is:

```
tracert [-options] host
```

Refer to the **tracert**(1Mtcp) manual pages for more information.

AV 5-18



5.4.3 The `rtquery(1Mtcp)` command

Use the `rtquery(1Mtcp)` command:

- To ask a network routing daemon using the RIP protocol for its routing table. For example, in AV 5-18, to find out routing information on the router *spider1* to the machine *firefly*, enter:

```
rtquery -r spider1 firefly
```

- To turn tracing on or off in a **routed** daemon

For this to be useful, the `routed` daemon needs to be started with a trace file (using the **-T** option). This can be done by modifying the line that starts **routed** in the `/etc/inet/config` file.

For example, after tracing is turned on, **routed** running on the local machine can be forced to dump its current routes to the trace file using this command:

```
rtquery -t dump localhost
```

A good use for this command is for the administrator of a large network to turn tracing on at the startup of all routers running **routed**, then sending them the **dump** command whenever there appears to be a problem with the routes on a given machine.

5.4.4 The `ripquery(1Mtcp)` command

The `ripquery(1Mtcp)` command requests all routes known by a RIP gateway by sending a RIP REQUEST or POLL command. The routing information in any routing packets returned is displayed numerically and symbolically.

This command is intended to be used as a tool for debugging routers, not for network management. SNMP is the preferred protocol for network management.

5.5 Lesson Summary

Routers allow you to connect networks and subnets together. This lesson described how to install additional network interfaces with the **Network Interface Manager** and then configure the software to allow IP packets to be sent from one network to another. Finally, several diagnostic tools were described to help you observe the routing activity on your system.

Review questions

1. Which kernel parameter is used to enable or disable kernel routing? How does it work?
2. What is the purpose of the `/etc/inet/gateways` files?
3. What are the steps a system takes to route an IP packet to a system not directly connected to the local network segment?
4. What is the kernel routing table? How does it get updated?
5. What are the three routing daemons supported by UnixWare 7?
6. Which utilities can you use to monitor routing on your system?

5.6 Lab: Configuring a Router

The purpose of this lab is to modify the classroom network to include one or more routers. You will also configure and test their associated routes.

5.6.1 Basic router setup

Your instructor will provide you with the information on how to modify the classroom network's topology to include one or more routers. Be sure to write down information relating to the reconfiguration of your host including:

- The new addresses necessary unless you are subnetting an existing network.
- The changes to the protocol parameters of the existing network adapter, for example its netmask.
- The IP address of the new network adapter your host will have if it is going to become a router.

After gathering the appropriate information, modify your host to implement the instructor's plan.

Procedure

You must be logged in as **root** to perform this lab. You will use procedures from labs in previous lessons.

If you are configuring your host as a router:

- Step 1. Follow the steps for adding a new network adapter in Section 3.1.2.1, “Adding a network adapter,” on page 3-9. Indicate that you want to configure your system as a gateway (refer to the procedure in Section 5.1.1, “Installing additional network adapters,” on page 5-7).

If you need to reconfigure the protocol parameters of your existing network adapter:

- Step 2. Follow the steps for modifying a network adapter’s protocol parameters listed in Section 3.3.2, “Modifying and removing components,” on page 3-21.

If you are using */etc/hosts* to resolve host names to IP addresses in your classroom network (your instructor will tell you):

- Step 3. Modify */etc/hosts* to reflect the changes in the classroom network’s topology. This can involve changing some entries as well as adding entries for the new network adapters added to routers.

You can perform this step by either using **vi**(1) on the */etc/hosts* file or by using the **Client Manager**. Refer to Section 3.4, “Configuring the */etc/hosts* File,” on page 3-23 for details.

- Step 4. Test your connectivity by using the tools described in Section 5.4, “Troubleshooting Routes,” on page 5-37. (HINT: Do not forget to use **netstat -r**).

Remember that the router must be configured correctly and running in multi-user mode before you will be able to reach systems in other networks.

5.6.2 The `ipforwarding` parameter

The objective of this lab is to observe the effects of the `ipforwarding` kernel parameter. You will observe the effects by:

- Watching packets flow normally through a router
- Turning off the `ipforwarding` parameter
- Observing that packets are no longer flowing through the router
- Turning the `ipforwarding` parameter back on
- Observing that packets now flow through the router again

Before continuing, verify that everyone in the class is at this point in the lab and that each system can connect to every other system.

Procedure

NOTE: The procedure outlined below uses the command line. You can also use the **scoadmin(1M) Network Configuration Manager**.

- Step 1. On the router systems only, use **inconfig(1Mtcp)** to verify the value of **ipforwarding** is 1.

inconfig -v ipforwarding

- Step 2. Observe that packets flow through the router by pinging from a host on one side of the router to a host on the other side of the router.

- Step 3. Use **inconfig** to change **ipforwarding** to 0.

inconfig -v ipforwarding 0

- Step 4. Test connectivity from non-routers and routers to other systems by using the **ping(1Mtcp)** and **rlogin(1tcp)** commands:

- On non-routers, are you still able to connect to systems in other networks? Why or why not?
- On the routers, are you still able to connect to systems in other networks? Why or why not?

- Step 5. For the remainder of the class, the **ipforwarding** kernel parameter must be set to 1. Reset **ipforwarding** to 1 with **inconfig**.

inconfig -v ipforwarding 1

- Step 6. Observe that packets flow through the router by pinging from a host on one side of the router to a host on the other side of the router.

5.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

gated (1Mtcp)	Is a routing daemon that handles multiple routing protocols
inconfig (1Mtcp)	Is a configuration utility for the default kernel parameters used by the TCP/IP protocol suite
mrouted (1Mtcp)	Creates a multicast tunnel through unicast routers
netcfg (1M)	Configures networking products and can be invoked as the Network Configuration Manager
netstat (1Mtcp)	Examines currently configured network interfaces and other network-related data structures, including routing table information
ping (1Mtcp)	Verifies connectivity of the hosts on a network
ripquery (1Mtcp)	Requests all routes known by a RIP gateway
route (1Mtcp)	Manually manipulates the routing tables
routed (1Mtcp)	Is a routing daemon that implements the Routing Information Protocol (RIP)
rtquery (1Mtcp)	Requests information from a routing daemon running the Routing Information Protocol (RIP)
traceroute (1Mtcp)	Traces and displays the route an IP packet follows to another Internet host

Relevant files

<i>/etc/inet/inet.dfl</i>	Configuration file for the inconfig (1Mtcp) command. The format of and parameters contained in this file are documented in the inconfig (1Mtcp) manual page.
<i>/etc/inet/gateways</i>	Configuration file for routed (1Mtcp) that is used to initialize routes to distant networks. The format of this file is documented in the routed (1Mtcp) manual page.
<i>/etc/inet/config</i>	Configuration file for TCP/IP daemons documented in the <i>config</i> (4tcp) manual page.
<i>/etc/inet/mrouted.conf</i>	Configuration file for mrouted (1Mtcp). The contents and format of this file are documented in the mrouted (1Mtcp) manual page.

5.8 Answers to Review Questions

1. The `ipforwarding` kernel parameter enables or disables kernel routing. When it is set to one (1), the system routes IP packets between any of its own network interface cards. If it is set to zero (0) the system will not route packets between any interfaces.
2. The **routed**(1Mtcp) daemon uses the `/etc/inet/gateways` file to initialize static routes to distant networks.
3. A system routes an IP packet to a system not directly connected to the local network in the following manner: the IP layer receives a packet addressed to the remote host and it consults its kernel routing table. The local host applies each netmask to the destination IP address until it finds a match with the destination address. If no match is found, the default entry is used; otherwise, it uses the gateway and interface fields of the entry and then addresses the packet to the specified gateway machine. When the gateway receives the message, it reads the ultimate destination IP address. Because the address is not its own and because it is configured as a router, it consults its own kernel routing table. There, it will find that the ultimate destination can be reached on the local gateway.
4. The routing table is a kernel data structure that contains the following network information: destination network IP address, netmask, next hop (gateway), and the identity of the network interface through which the packet is sent to reach the next hop. The routing table is updated automatically when running the **Network Configuration Manager** to configure or remove a network interface; manually, using the **route**(1Mtcp) command; and dynamically, using specialized routing daemons running on the system.
5. The three routing daemons supported by UnixWare 7 are **mrouted**(1Mtcp), **routed**(1Mtcp), and **gated**(1Mtcp).
6. You can use the **netstat**(1Mtcp), **traceroute**(1Mtcp), **ping**(1Mtcp), **rtquery**(1Mtcp), and **ripquery**(1Mtcp) utilities to monitor routing on your system.

6

Configuring WAN Connections

AV 6-1

The slide is titled "Configuring WAN Connections" in a blue-bordered box at the top. Below the title, there is a bulleted list of four items. At the bottom of the slide, there is a blue footer bar containing the text "GNAD0601" on the left and "© The Santa Cruz Operation, Inc." on the right.

- WAN interface types
- WAN protocols
- Configuring the point to point protocol (PPP)
- Troubleshooting PPP

GNAD0601 © The Santa Cruz Operation, Inc.

Lesson 6

Configuring WAN Connections

Introduction

Previous lessons have assumed that all network connections are made by Ethernet cards and have been LAN-centric.

Your networking requirements can also include connecting to the Internet, or connecting multiple LANs to form a wide area network (WAN).

You can complete these requirements using SCO PPP (Point-to-Point Protocol).

Objectives

This lesson discusses how PPP connects LANs with other LANs, Hosts, and the Internet. It also discusses other networking interfaces that you can use as a network administrator. By the end of this lesson, you will be able to:

- List the interfaces that connect LANs to one another and the Internet
- Configure the various PPP interfaces and associated support files
- Connect to an ISP with PPP
- Troubleshoot PPP

6.1 WAN Interface Types

Wide Area Networks (WANs) connect host, or LANs that are separated by distances greater than those supported by LAN technology. Several interfaces developed to connect WANs include:

- Modems
- ISDN
- DSL
- Leased Lines
- Frame Relay
- Cable modems

6.1.1 Modems

The term “modem” is derived from the terms *modulator* and *demodulator*. A modem is a device that:

- Converts binary data into an audio signal for transmission over a phone line (modulation)
- Receives an audio signal and converts it back to binary data (demodulation)

Three important factors to consider when choosing a modem are speed, latency, and the standards supported by the modem.

Speed can be measured by either *baud rate* or *bps* (bits per second). Baud rate is:

- The rate at which the transmitted signal changes per second, *not* a measure of transmission speed since multiple bits can be encoded into each signal.
- Often used incorrectly as a synonym for bps which is a measure of transmission speed.

Modem Modulation Standards

Modulation	Data Rate (bits/second)	Comments
Bell 103	300	
Bell 212A	1,200	
V.22	1,200	
V.22bis	2,400	
V.32	9,600	
V.32bis	14,400	
V.terbo	19,200	Pre-V.34 modulation with fewer features.
V.FC	28,800	
V.34	28,800	
V.34bis	33,600	
K56	56,000*	Proprietary modulation used by Rockwell & Lucent
X2	56,000*	Proprietary modulation used by 3COM
V.90	56,000*	New ITU Standard. Previously referred to as V.pcm.

(*) Achieve 56,000 bps only on downloads from specially configured providers, otherwise rate is 33,600 bps. The different modulations are incompatible.

bps is:

- A measure of the data transmission rate, in bits of data per second
- Determined by the modulation standard used

To convert to bytes per second, you must understand that there are typically eight data bits and two overhead bits per byte. Therefore dividing *bps* by 10 will yield the rate in bytes per second.

Latency is the delay incurred due to the modem processing the data prior to its transmitting. A high latency time does not lower the amount of data that can be transmitted per second but causes the data to delay which might degrade performance. Some lower quality modems have excessive latency times which can cause performance loss.

Standards ensure that modems from different manufactures can communicate effectively. Several standards have been set by organizations such as the International Telecommunications Union, Telecommunication Standardization Sector (ITU-T). Standards control and include:

- Modulation—The modulation standard being used determines the highest data transmission rate possible. Line noise can cause actual data transmission to go slower. See AV 6-2 for a list of modulation standards.

AV 6-3

Other Modem Standards	
Standard	Comments
V.42 error detection and correction	Specifies LAPM error correction and can fall back to MNP 4 for compatibility.
V.42bis data compression	A ITU-T data compression standard. Most modems with V.42bis can fall back to MNP Class
V.17 fax modulation	Sends and receives faxes at 14,400 bps or slower if necessary.

GNAD0603 © The Santa Cruz Operation, Inc.

- Error detection and correction—Allows the modem to detect and correct errors in data transmission. AV 6-3 shows the current standard.
- Compression—Causes the data to be compressed by the modem before transmission to increase data throughput. AV 6-3 shows the current standard.
- Fax modulation—Enables the modem to send and receive facsimiles. AV 6-3 shows the current standard.

Another possible source of confusion regarding speeds involves the connection between the computer and the modem:

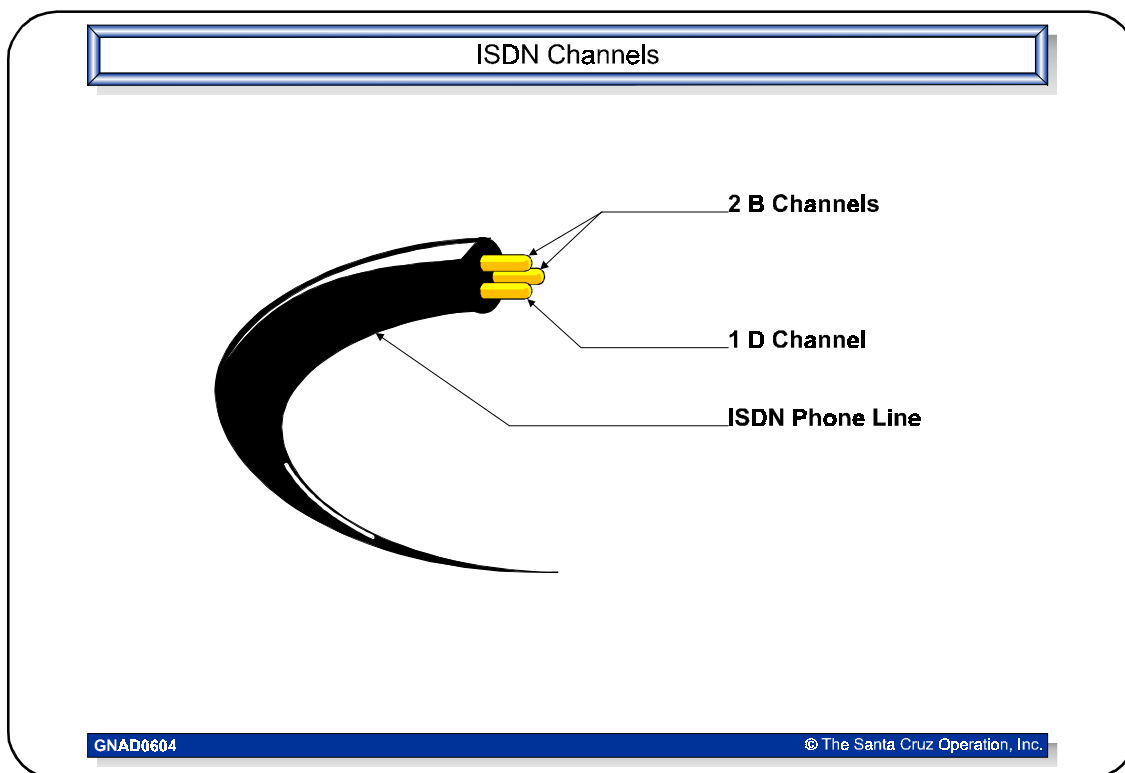
- The modems typically negotiate the speed at which they will talk to each other.
- The speed the computer talks to the modem can also be configured.

You will want to configure this to be higher than the modem's maximum connection speed to accommodate any compression the modem might be using.

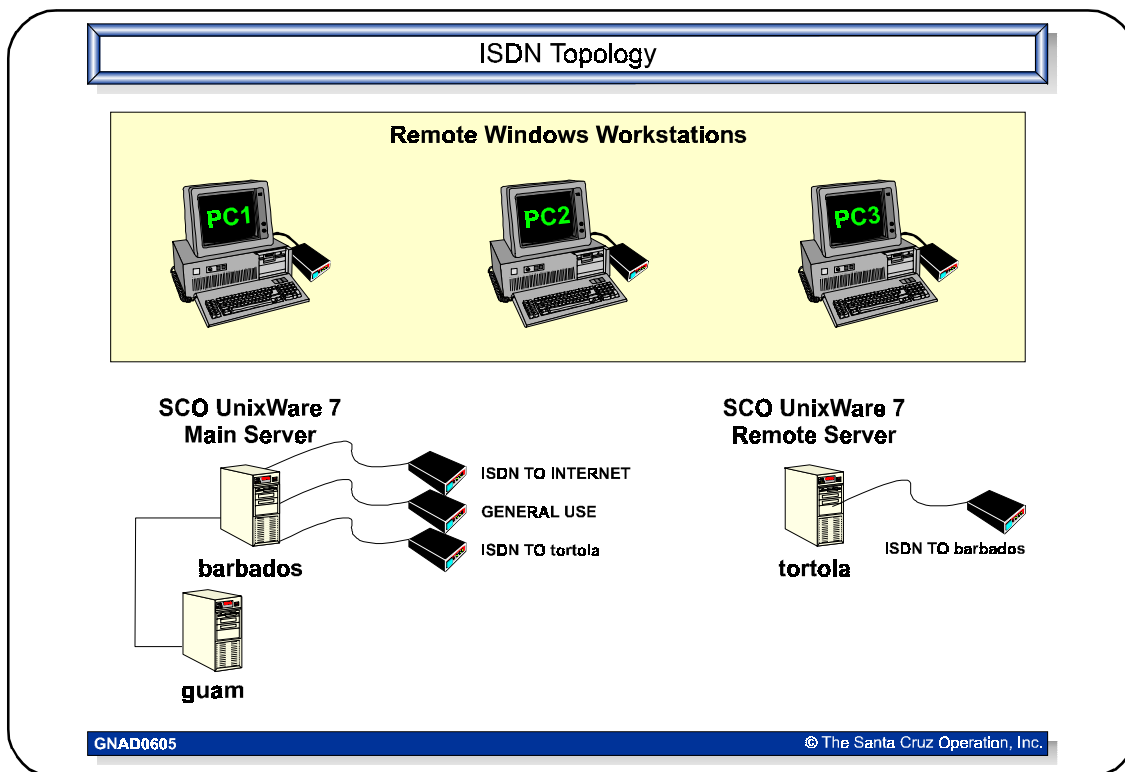
When contact is initiated between modems they:

1. Negotiate which standards they can both support.
2. Determine the speed the line conditions can support.
3. If there is excessive line noise or if the other modem cannot support the highest standard, the modem typically changes to a lower standard or a slower speed and tries again. This procedure is repeated until an appropriate standard is reached.

AV 6-4



AV 6-5



6.1.2 ISDN

Integrated Services Digital Network (ISDN) is a digital WAN product that uses existing phone lines to allow digital transmission of voice, data, video. It is faster and more efficient than modems, which must convert digital signals into sound and back again.

An ISDN setup consists of two types of channels, *bearer* and *data*:

- ISDN Bearer (B) channels transmit data at up to 64Kbps each
- ISDN Data (D) channels carry control information for the phone company, such as call setup information.

Three common ISDN devices in use are:

- An external ISDN router that has an Ethernet or Token Ring interface to connect directly to the LAN.
- An external ISDN adapter that connects through a serial port as a modem does. To take advantage of all of the ISDN bandwidth, you must configure the serial port to run at least as fast as the ISDN adapter.
- An internal ISDN adapter board that attaches to the bus (for example, PCI or ISA) inside your existing server. This overcomes the serial port problems with the external adapters.

These devices are configured to bring up a PPP connection with remote ISDN devices. They can be used to connect two LANs or connect with an Internet Service Provider (ISP).

AV 6-5 shows a possible ISDN topology. For more information on ISDN, see <http://www.acc.com/White/isdn.html>.

6.1.3 Other digital subscriber line technologies

There are several digital WAN technologies that are not directly supported by UnixWare 7:

- DSL—Digital Subscriber Line is an umbrella term for a family of similar technologies that use frequency ranges above standard voice communications to carry data.
- ADSL—Asymmetrical Digital Subscriber Line technology maintains a permanent connection, so it can be used for providing Internet services such as HTTP, FTP, and e-mail.
- VDSL—Very high data rate Digital Subscriber Line is similar to ADSL but provides faster data rates at shorter cable lengths.
- HDSL—High data rate Digital Subscriber Line has symmetrical transmission rates; uploads and downloads proceed at the same rate.
- SDSL—Single line Digital Subscriber Line is similar to HDSL but uses a single POTS (plain old telephone service) line for transmission making it more suitable for many subscriber's premises.

More information about these services can be found at various sites on the Web, such as <http://www.adsl.com>.

AV 6-6

Leased Line Speeds in North America and Japan

Common Name	Official Name	Description	Throughput
T1	DS1	24 64Kbps voice lines multiplexed together	1.5Mbps
T2	DS2	4 T1 lines multiplexed together	6.3Mbps
T3	DS3	7 T2 lines multiplexed together	45Mbps
T4	DS4	6 T3 lines multiplexed together	274Mbps
Fractional T1	FT1	Any integer multiple, up to 23, of 64Kbps voice lines multiplexed together	64Kbs * the number of lines
Fractional T2	FT3	Any integer multiple, up to 27, of 1.5Mbps T1 lines multiplexed together	1.5Mbs * the number of lines

GNAD0606

© The Santa Cruz Operation, Inc.

AV 6-7

Leased Line Speeds Elsewhere in the World

Official Name	Description	Throughput
E1	30 64Kbps voice lines multiplexed together	2.048Mbps
E2	4 E1 lines multiplexed together	8.448Mbps
E3	4 E2 lines multiplexed together	34.368Mbps
E4	4 E3 lines multiplexed together	139.264Mbps
E5	4 E4 lines multiplexed together	565.148Mbps
Fractional E1	Any integer multiple, up to 29, of 64Kbps voice lines multiplexed together	64Kbs * the number of lines
Fractional E2	Any integer multiple, up to 3, of E1 lines multiplexed together	2.048Mbs * the number of lines

GNAD0607

© The Santa Cruz Operation, Inc.

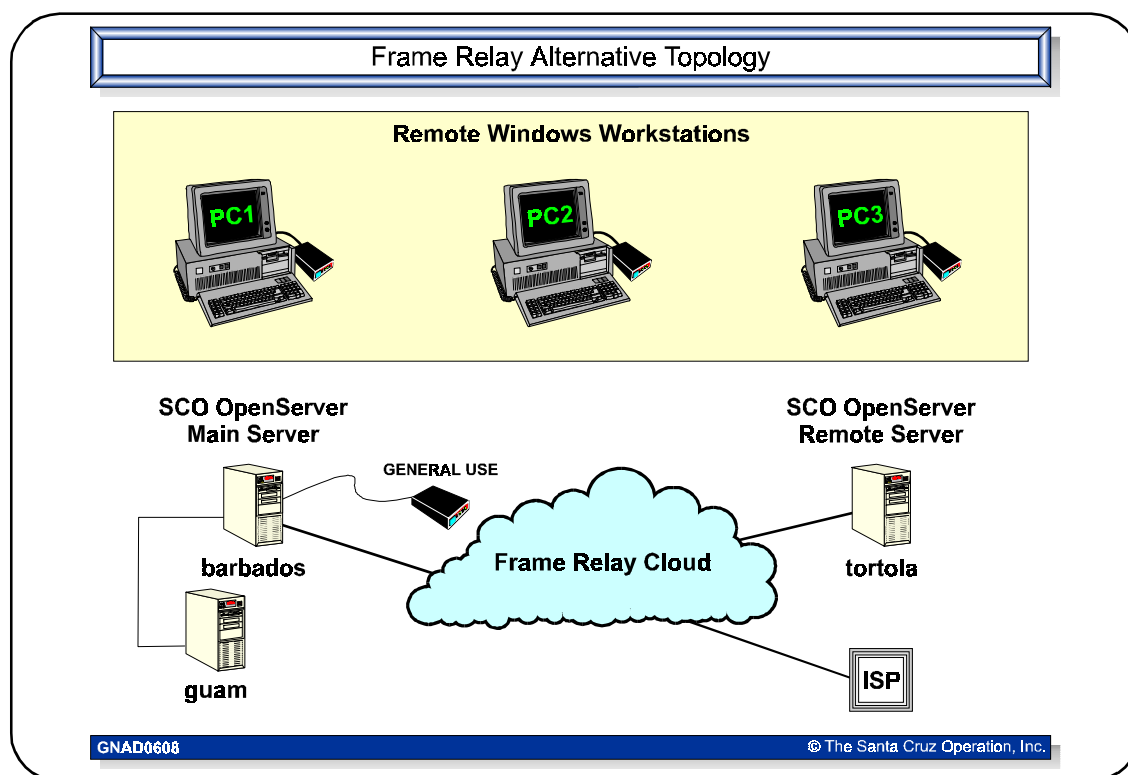
6.1.4 Leased lines

Leased telephone lines are point to point connections which:

- Require special telephone equipment called a CSU/DSU. This is:
 - Used to connect to the phone system
 - Usually connected to a router of some kind, or a dedicated board in a host computer
- Can use either SLIP or PPP
- Can have a considerable cost
- Come in various speeds from 64Kbps to 274Mbps as described in AV 6-6.

The Internet's backbone is comprised of special places known as network access points (NAPs). NAPs are connected with leased lines running at T3 speeds or better.

AV 6-8



6.1.5 Frame relay

Frame relay is similar to ISDN because:

- It goes through the phone system.
- It is purely digital.
- It has a variable throughput, ranging from 56Kbps to 1.5Mbps in 64Kbps increments.

It is different than ISDN because:

- It is not a point to point connection.
- It is a dedicated connection, not switched or dialed.
- It sends packets from your interface which are destined for some remote frame relay interface via a frame relay *cloud* implemented by the phone company. The frame relay *cloud* behaves very much like the Internet in that packets are routed within the cloud to the destination frame relay interface, typically one of your other locations.
- It requires special hardware from the phone company connected to a dedicated phone line. This hardware typically is connected to a dedicated router or a frame relay board.
- It is usually more expensive to set up and is charged a higher monthly fee than ISDN.
- It is usually charged at a flat rate.
- It is more versatile than ISDN because a single frame relay interface can be used to send data to any of your other frame relay interfaces packet-by-packet. ISDN requires separate connections to perform the same task.
- It is a standard of the T1S1 committee of ANSI and ITU-T.

6.1.6 Cable modems

Cable modems:

- Uses existing coax wiring that is currently being used for cable television to also carry data communications
- Allows regular television programming and digital data to be transmitted simultaneously

- Currently has no industry wide standards so performance and operation of cable modems vary between different implementations:

Because bandwidth is shared with other users on a segment in much the same way ethernet bandwidth is shared:

- Performance at a site can vary greatly depending on the activities of other users.
- It might be an unsuitable choice for any user requiring consistent bandwidth.
- Packet sniffing could pose security risks.

6.2 WAN Protocols

The physical differences between WAN and LAN hardware requires that WAN connections use separate protocols to handle the issues related to WAN hardware.

The original Internet consisted of several LANs interconnected using dedicated phone lines, called *leased lines*, running at speeds from 9600–56Kbps. This was expensive.

As more sites were added to the Internet and the Internet community thought about opening access to the public at large, it was clear that access would need to be made available through something other than leased lines.

6.2.1 Serial Line Internet Protocol (SLIP)

The original Serial Line Internet Protocol (SLIP) was invented to allow TCP/IP to run over serial lines, much like PPP, except that SLIP was originally designed to work over dedicated serial lines with hardware flow control.

There was no concept of a connection timing out and being dropped, or dialing a modem.

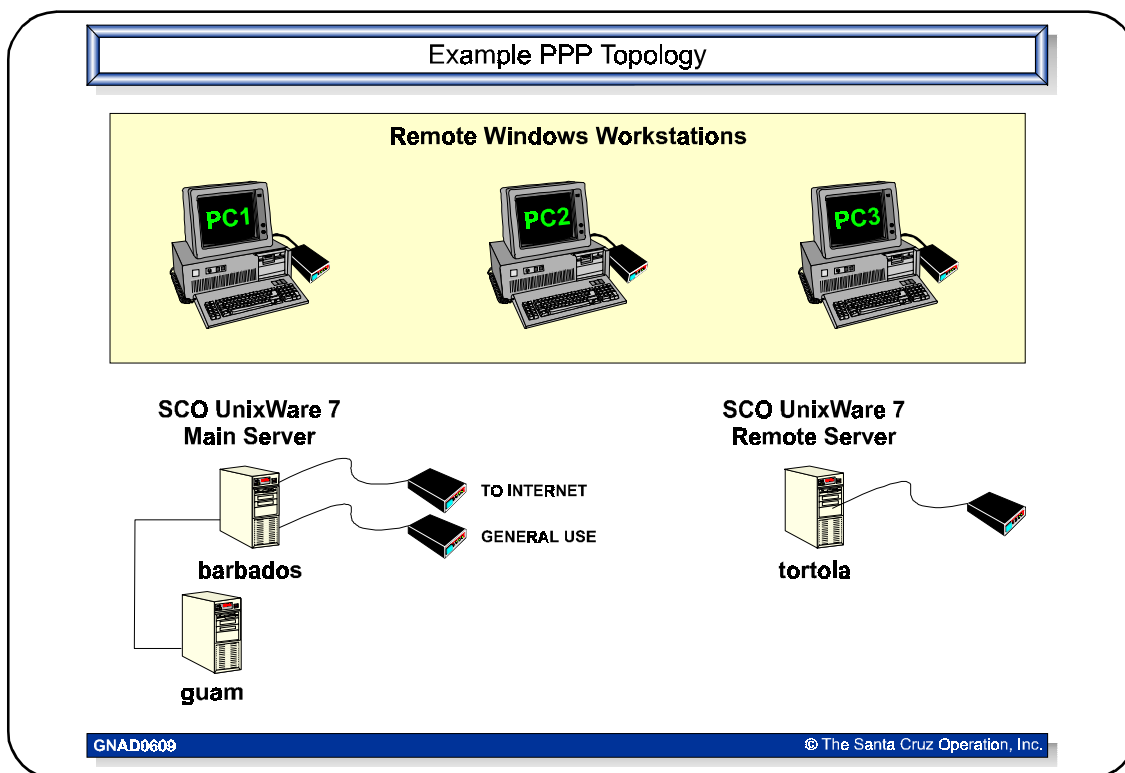
Modern SLIP implementations can have many PPP features, including:

- PAP and CHAP authentication
- Modem dialing
- Header compression
- Connection time-out

Note: The UnixWare 7 implementation of SLIP does not support PAP/CHAP, compression, or connection time-out. It does support modem dialing using UUCP but not on demand.

Because it was designed to work over dedicated serial lines with hardware flow control, SLIP has no need for control-character escapes as PPP does. Consequently, some modern SLIP implementations might be faster than some PPP implementations.

AV 6-9



6.2.2 Point-to-Point Protocol (PPP)

The most popular protocol for WAN interfaces is the Point-to-Point Protocol (PPP). PPP provides TCP/IP services over WAN connections:

- Allowing the transmission of packets of any network protocol (for example, IP and IPX) over serial or phone lines
- Allowing negotiation of configurable options between the two peers, unlike SLIP which only allows a fixed configuration
- Dialing the modem to make outgoing connection requests and answering the modem for incoming connection requests
- Monitoring the line and dropping the connection if no packets are sent for some period of time

While many WAN devices (such as ASCEND or CISCO routers) support IPX and other protocols over PPP, UnixWare 7 only supports IP over PPP.

6.2.2.1 PPP bundles and link devices

UnixWare 7 defines a PPP connection as a *bundle*. A bundle is a set of configuration parameters grouped together to describe how a connection is made.

There are three types of bundles:

Outgoing Specifies the remote system name to contacted

Incoming Specifies the local user name used by the remote user

Bi-directional Specifies both incoming and outgoing. However, bi-directional bundles are less common because incoming and outgoing networking parameters are often different, and therefore must be configured in different bundles.

A bundle includes configuration settings for IP parameters, such as IP addresses and netmasks. It also includes *link devices*, which define the physical device used to make a connection, such as a modem or ISDN adapter.

In the simplest case, you might have one bundle for an incoming or outgoing connection which includes only one link device. However, UnixWare 7 also supports *multilink* PPP, which allows a single bundle to use multiple link (physical) devices.

6.2.2.2 Multilink PPP

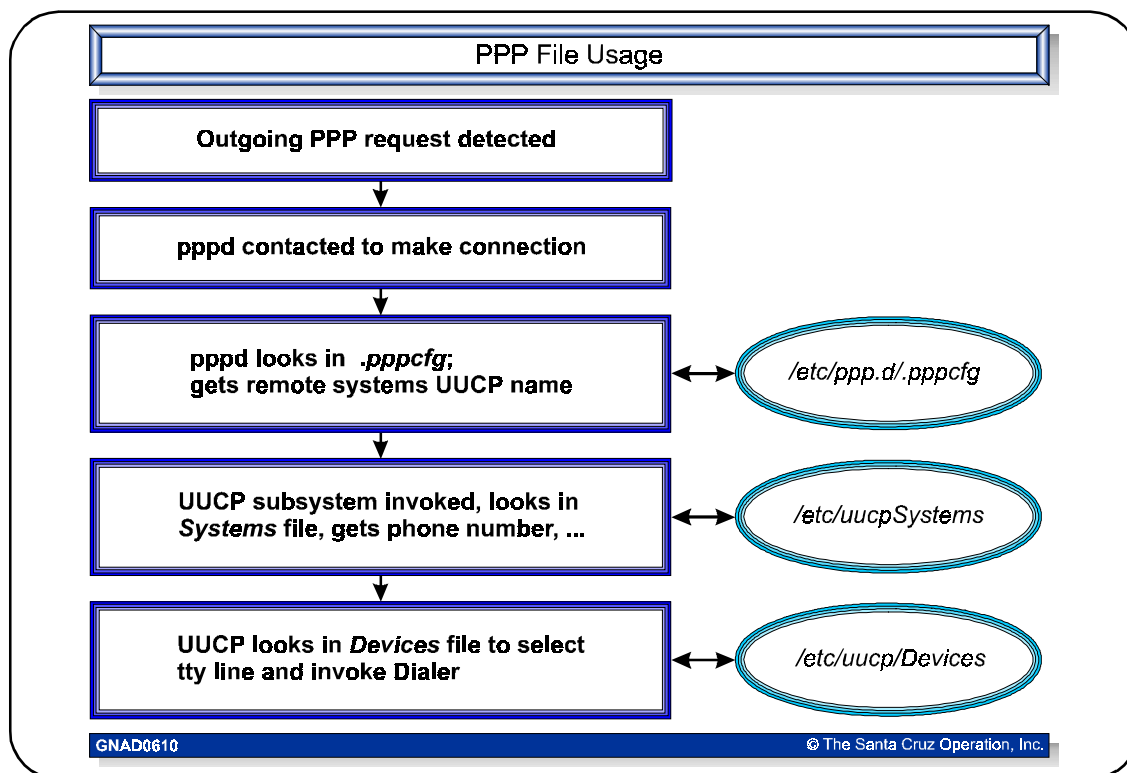
Multilink PPP allows the use of multiple devices to build a connection between two systems. For example, several ISDN adapters or several modems and phone lines can be used together to form a single higher bandwidth connection.

Multilink PPP:

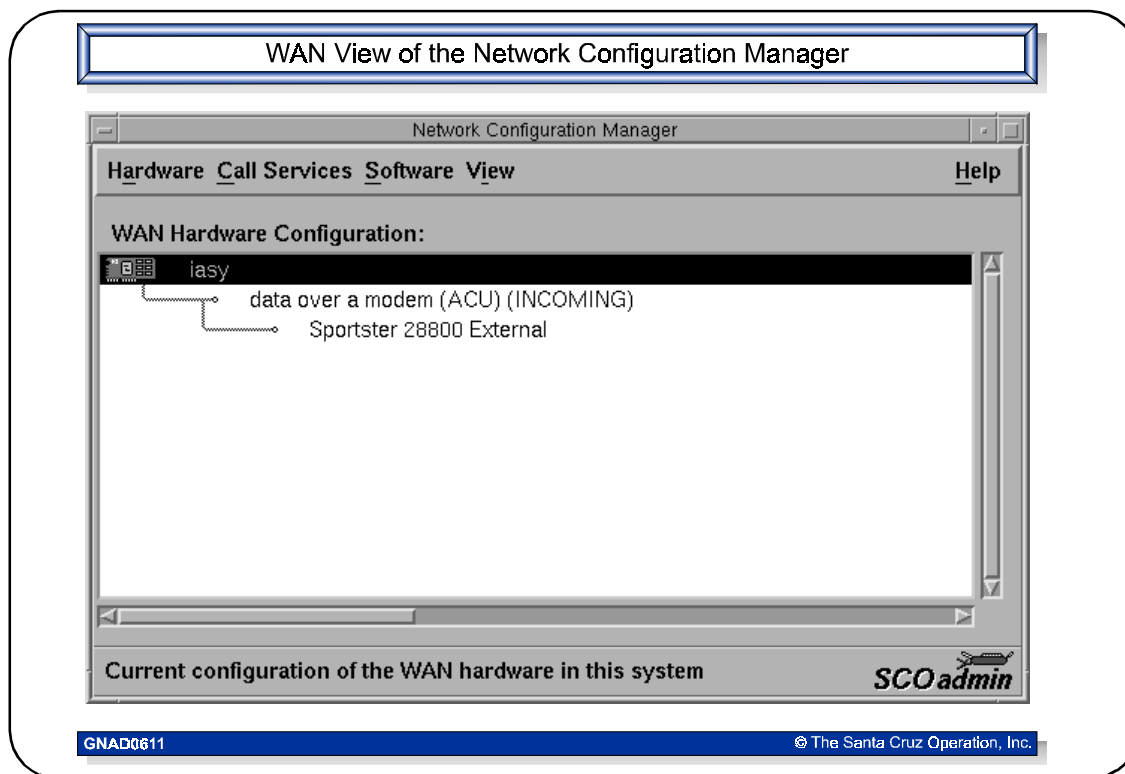
- Increases WAN bandwidth by sending packets in parallel over the multiple links
- Must be supported by both ends of the connection
- Typically costs more because it uses more phone lines and more of an ISP's bandwidth
- Supports *bandwidth on demand*—the system can be configured to add or drop links as required by increases or decreases in traffic.

Note that multilink PPP is not recommended over analog modems because of the variable line quality of analog phone lines.

AV 6-10



AV 6-11



6.3 Configuring PPP Connections

The **Network Configuration Manager** is used to configure PPP connections. The basic procedure involves two stages:

1. Configure WAN hardware. This stage configures the physical link device characteristics for the modem and serial port. It also configures UUCP to dial the modem (for outgoing connections).
2. Add incoming or outgoing bundles (connections). This configures PPP to use one or more link devices, configures the remote system name (for outgoing connections), sets IP parameters, and configures other PPP parameters.

When you complete each phase, the **Network Configuration Manager** modifies the applicable configuration files. The relationship between these files is illustrated in AV 6-11.

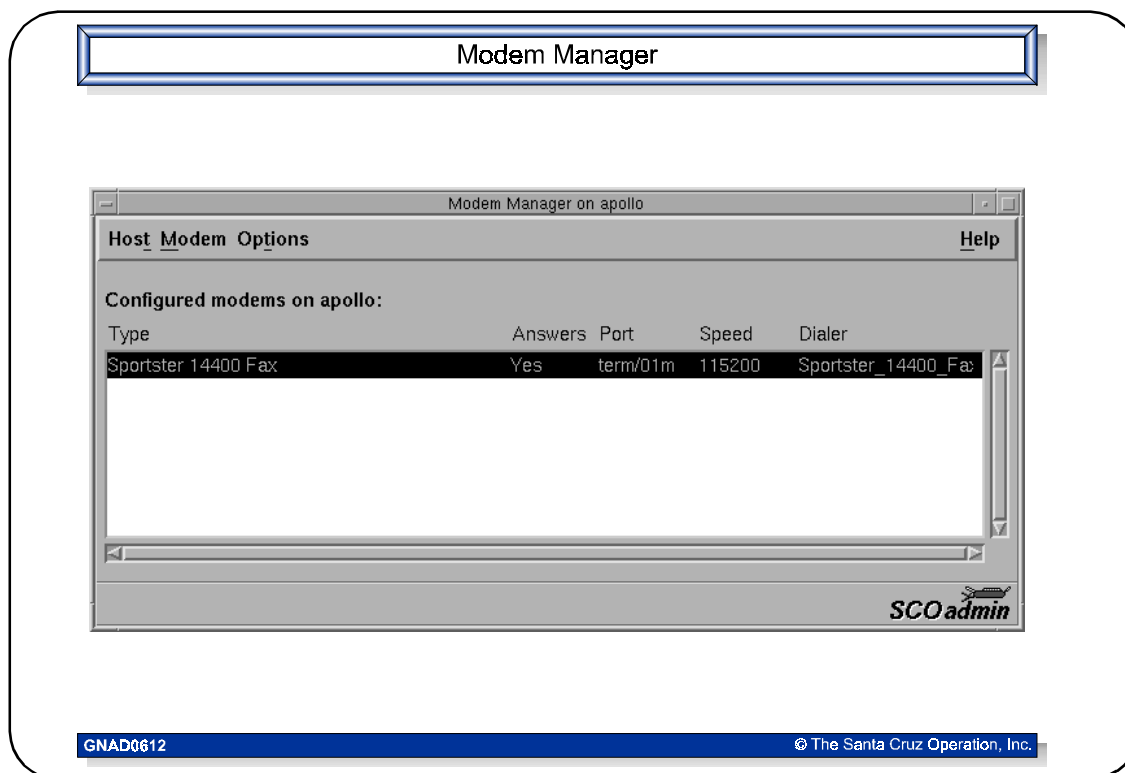
6.3.1 Configuring devices for WAN connections

Add or modify link devices using the **Hardware** menu of the **Network Configuration Manager**. Operations in this menu modify the `/etc/uucp/Devices` file. See *Devices(4bnu)* for more information on the use and structure of this file.

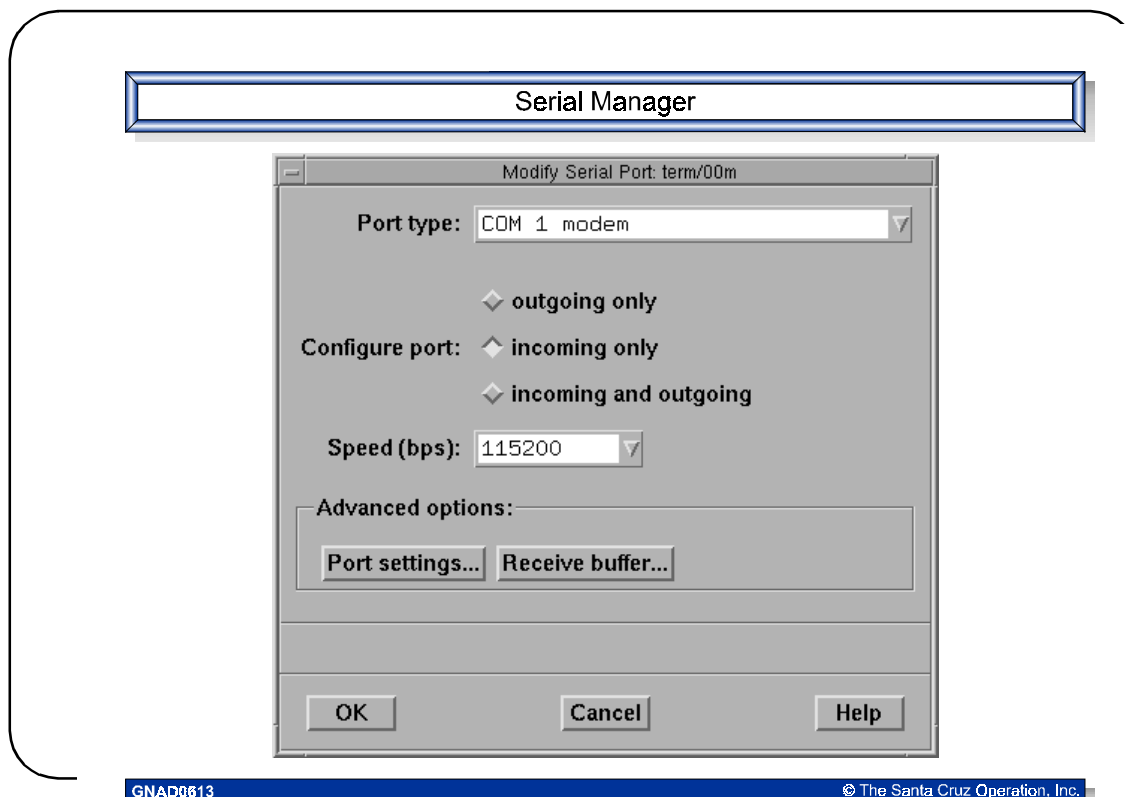
To add a modem for a WAN:

- Step 1. Run **SCOadmin**.
- Step 2. Select **Networking** ⇒ **Network Configuration Manager**.
- Step 3. Select **View** ⇒ **WAN**.
- Step 4. Select **Hardware** ⇒ **Add new WAN Device** ⇒ **Modem**. This starts the **Modem Manager**. Modems already configured on the system are displayed.

AV 6-12



AV 6-13



Step 5. Select **Modem** ⇒ **Add**.

Step 6. Select **Manual Configuration**.

- Select the “Modem vendor” and “Modem model” from the lists.
- Select the port the modem uses in the “Modem port” field.
- Click on **OK**.

Step 7. Modify the serial port configuration. Select the modem you added in Step 6. Select **Modem** ⇒ **Modify**. Click on **Configure port** to start the **Serial Manager** shown in AV6-14. Set these options:

Port type	Selects what type of port this is. In this case, modem is the only appropriate choice.
Configure port	Select outgoing or incoming, but not both.
Speed(bps)	Sets the communication speed of this port. The speed this port can be set to depends on the UART chip used in your computer as well as what the modem can support. For outgoing only connections, this can be set to auto and the port configuration is detected and set automatically.

Port Settings	Selects the number of data bits and the type of parity used in serial communications over this port.
Receive Buffer	Sets the trigger level number of bytes that must be in the UART's buffer before an interrupt is sent to the CPU. A higher number increases system performance but also increases the risk of bytes being lost. The default of eight is always displayed despite the actual current level.

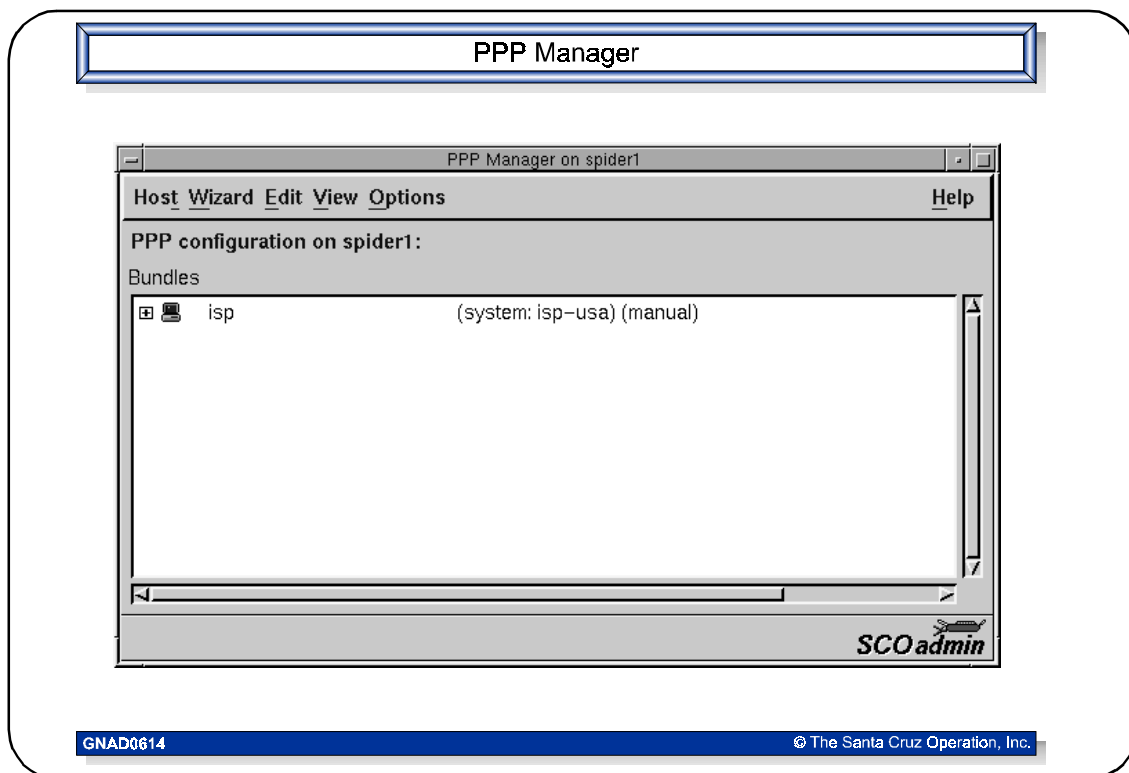
Step 8. Click on **OK** in the **Modify Serial Port** window.

Step 9. Click on **OK** in the **Modem Configuration** window.

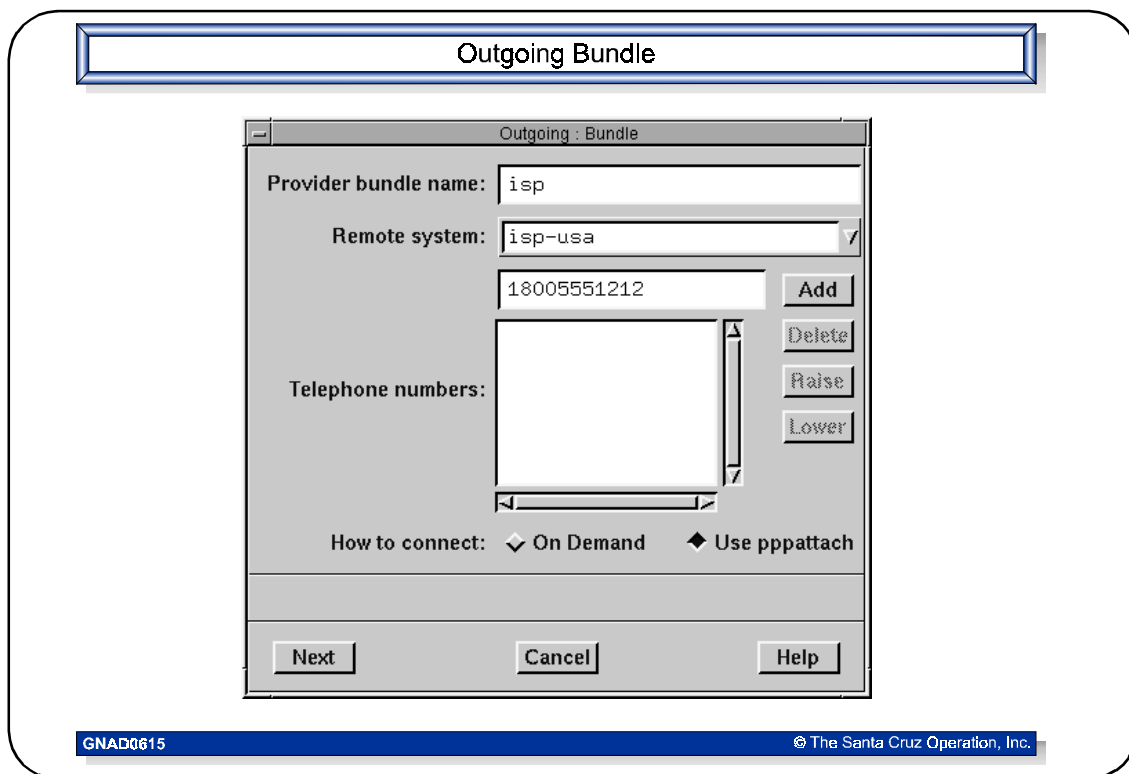
Exercise: Configuring a modem

Add a dial-out configuration for your system's modem or for a fictitious modem. Refer to the *Devices(4bnu)* manual page and examine the */etc/uucp/Devices* file for the configuration you just added.

AV 6-14



AV 6-15



6.3.2 Creating outgoing PPP connections

The **Network Configuration Manager** simplifies the task of creating outgoing PPP connections by providing a wizard to step through each configuration screen.

After you have added one or more WAN devices, configure a PPP outgoing link using these steps:

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select **View** ⇒ **WAN**.
- Step 3. Select **Software** ⇒ **Configure PPP**. This starts the **PPP Manager**.
- Step 4. Select **Wizard** ⇒ **Add** to add a new connection.
- Step 5. Select **Outgoing-Internet/Intranet**. Click on **Start**.
- Step 6. Complete the ISP configuration form, including:

Provider bundle name	Used to bring up this link with pppattach(1M) .
Remote system	Name used in the <i>/etc/uucp/Systems</i> file to refer to this connection. You can create a new entry or use the pull down menu to select an existing one.
Telephone numbers	Used to contact the remote system. Enter the number in the field under “Remote system” and click on Add . More than one phone number can be specified, in which case they are tried in the order listed.
How to connect	Specify whether the connection is brought up automatically when there are packets to cross the link or whether the link must be brought up manually using pppattach(1M) .

Click on **Next** when you are finished configuring the provider details.

AV 6-16

Outgoing PPP Authentication

Authentication protocol: ☒ None ☐ CHAP
☐ CHAP and PAP ☐ PAP

Local authentication name:

CHAP secret:

PAP password:

Next Back Cancel Help

GNAD0616 © The Santa Cruz Operation, Inc.

AV 6-17

Login Script Window

Use login script: ☒ Yes ☐ No

Login Script

1st Expect:		1st Send:	\n
2nd Expect:	in:--in:--	2nd Send:	myuserid
3rd Expect:	sword:	3rd Send:	mypassword
4th Expect:		4th Send:	
5th Expect:		5th Send:	

Next Back Cancel Help

GNAD0617 © The Santa Cruz Operation, Inc.

- Step 7. Complete the **PPP Authentication** configuration form, including:
- | | |
|---------------------------|---|
| Protocol | The protocols to respond to. Selecting None disables both. |
| Local authentication name | Name sent to the remote system to identify this system, analogous to a user name. |
| CHAP secret | Secret password to verify your authentication name for the CHAP protocol. |
| PAP secret | Secret password to verify your authentication name for the PAP protocol. |

Click on **Next** when you are finished configuring authentication.

- Step 8. Complete the **Login Script** configuration form.

This form specifies whether or not a login script is used to negotiate the connection. It also configures the script by specifying expect/send pairs.

The script shown in AV 6-17 specifies:

- Nothing expected, but send a `\r` (a carriage return)
- When `in:` is received, send the login name `myuserid`
- When `swor d:` is received, send the password `mypassword`

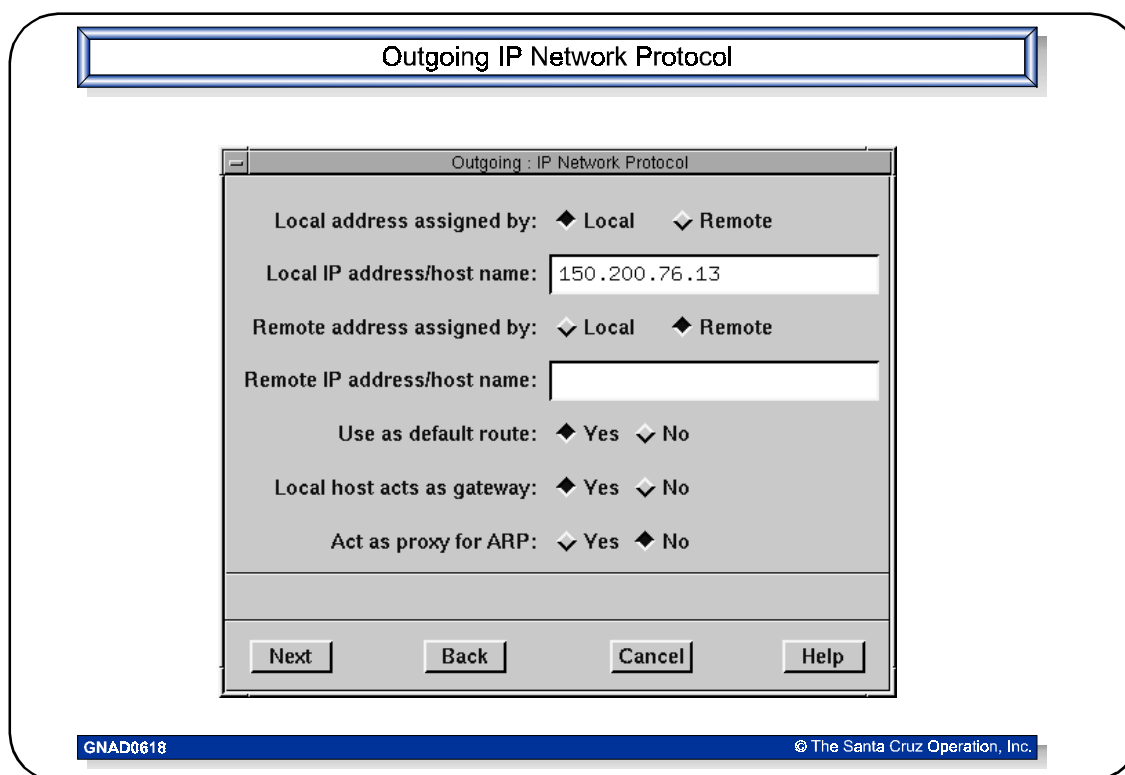
Note that the full expected words “login” and “password” were not used, avoiding any possible problems due to capitalization. Only enough characters to make the strings unique are necessary for the expect strings.

This window also modifies the `/etc/uucp/Systems` file.

If a password is included in the send scripts, it is stored *unencrypted* in this file. This can be a security concern.

Click on **Next** when you are finished configuring authentication.

AV 6-18



- Step 9. Complete the **IP Network Protocol** configuration form. Enter the TCP/IP parameters related to this PPP link:

Local address assigned by

Specifies whether this machine or the remote machine sets the IP address of the local side of the PPP link.

Local IP address/host name

Defines an IP address or resolvable name to be used for the local side of the PPP link.

Remote address assigned by

Specifies whether this machine or the remote machine sets the IP address of the remote side of the PPP link.

Remote IP address/host name

Sets an IP address or resolvable name to be used for the remote side of the PPP link.

Use as a default route

Specifies whether or not this link is to be used as a default route for IP packets. Packets with network addresses for which there are no configured routes are sent over the default route. See Section 5.2.6, “Default routes,” on page 5-21.

Local host acts as a gateway

Specifies whether this machine will forward packets between this link and other network interfaces. See Section 5.1.2, “Kernel parameters,” on page 5-9.

Act as a proxy for ARP

Use proxy ARP when the host at the end of the link is using an IP address that belongs to your LAN. If the connection is being used to join two distinct networks, or subnets, proxy ARP is not required.

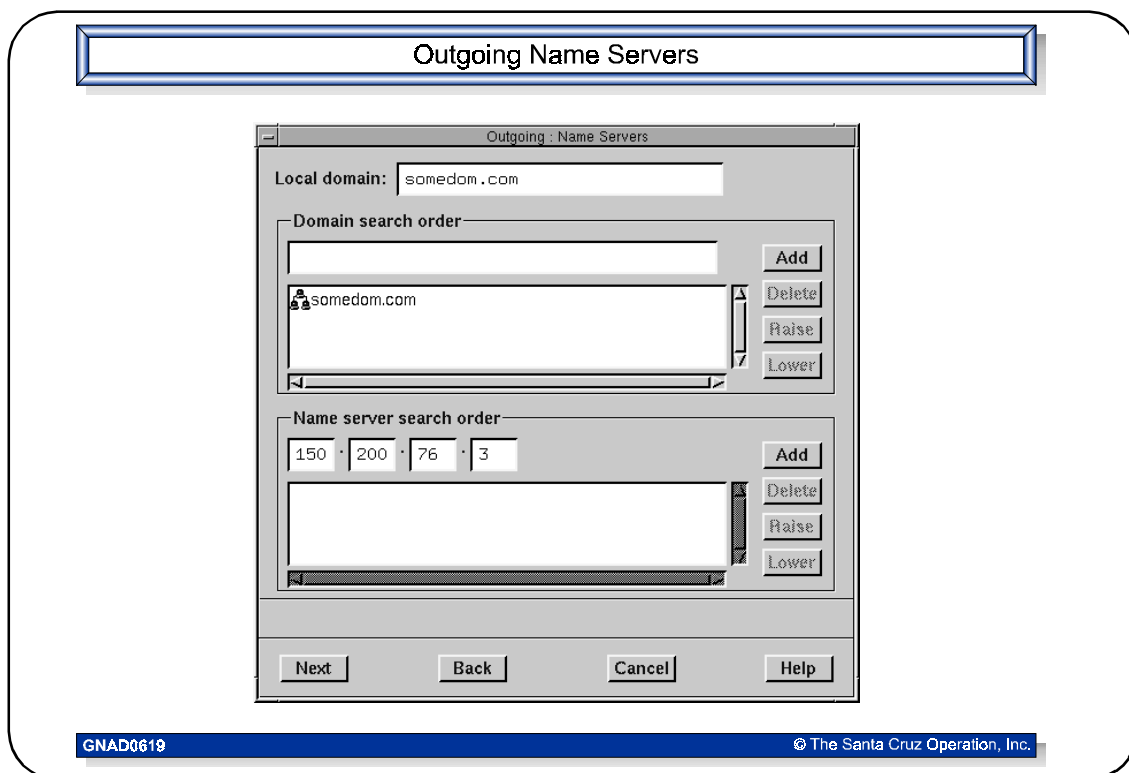
Click on **Next** when you are finished configuring IP parameters.

Note that if you are configuring the link to be brought up on demand, you must enter IP addresses in the local and remote address fields even if these are being set remotely.

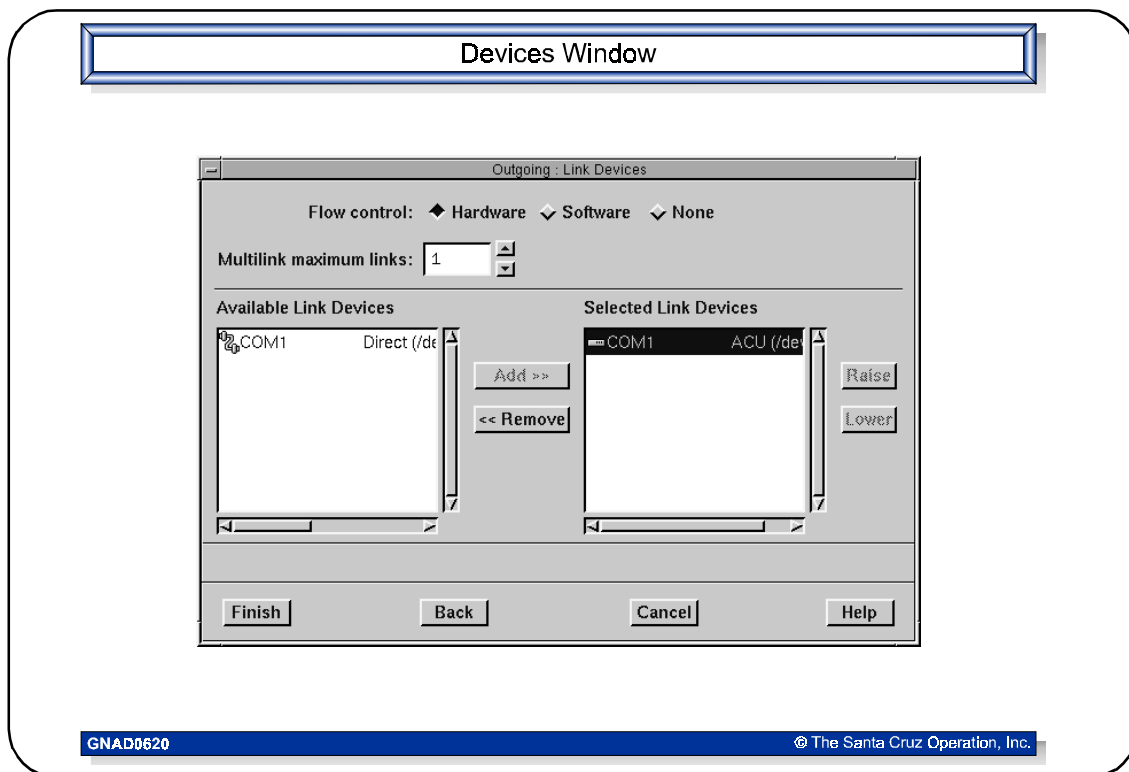
If set remotely, use sample addresses for these fields which will be reset upon link negotiation.

These sample addresses can be obtained from your service provider, or you can bring up the link manually and use the **ppptalk(1M)** command to detect the IP addresses set and reconfigure using those.

AV 6-19



AV 6-20



Step 10. Complete the **Name Servers** configuration form, including:

Local domain Specify the fully-qualified name for the domain to which this (local) system belongs.

Domain search order Enter the domain to search when you use a host name without a domain. Up to six domains can be entered.

Name server search order
Enter IP addresses of up to three servers to query.

Note that all network interfaces share the same name servers.

Click on **Add** for each domain or name server entered. Click on **Next** when you are finished configuring DNS.

Step 11. Complete the **Link Devices** configuration form. This lets you select which devices this bundle can use:

Flow Control Select the type of flow control to be used between the system and the device. Select from **Hardware** (RTS/CTS), **Software** (XON/XOFF), or None. Most devices will use **Hardware**. ISDN synchronous-mode devices do not use flow control.

Multilink maximum links

Select the maximum number of links to use in a connection to this host. To disable multilink, choose 1. To enable multilink, choose any number greater than 1.

Multilink must be supported by the remote host (your service provider). You might also need to configure multiple phone numbers in the **Outgoing Bundle** window if these extra links are not accessible by the same phone number.

Available Link Devices

Select the device(s) that will be used for the link. Click on **Add** to add the device to the “Selected devices” list. Repeat to add multiple devices.

If you select multiple devices, your connection can use any available device from the group.

- Step 12. Click on **Finish** to complete the outgoing PPP connection setup. Click on **OK** to confirm that the configuration completed successfully.
- Step 13. Select **Host** ⇒ **Exit** to exit the **PPP Manager**. The UUCP and PPP configuration files shown in AV 6-10 are now updated.

6.3.3 Using an outgoing PPP connection

The steps used to initiate an outgoing PPP connection depend on the connection method you selected:

On demand Use any networking command to send network traffic to the remote end of the connection, such as **ping**(1Mtcp) or **telnet**(1tcp). The connection is automatically made when it is needed.

Use **pppattach** Use the **pppattach**(1M) command to bring up the connection. The syntax is:

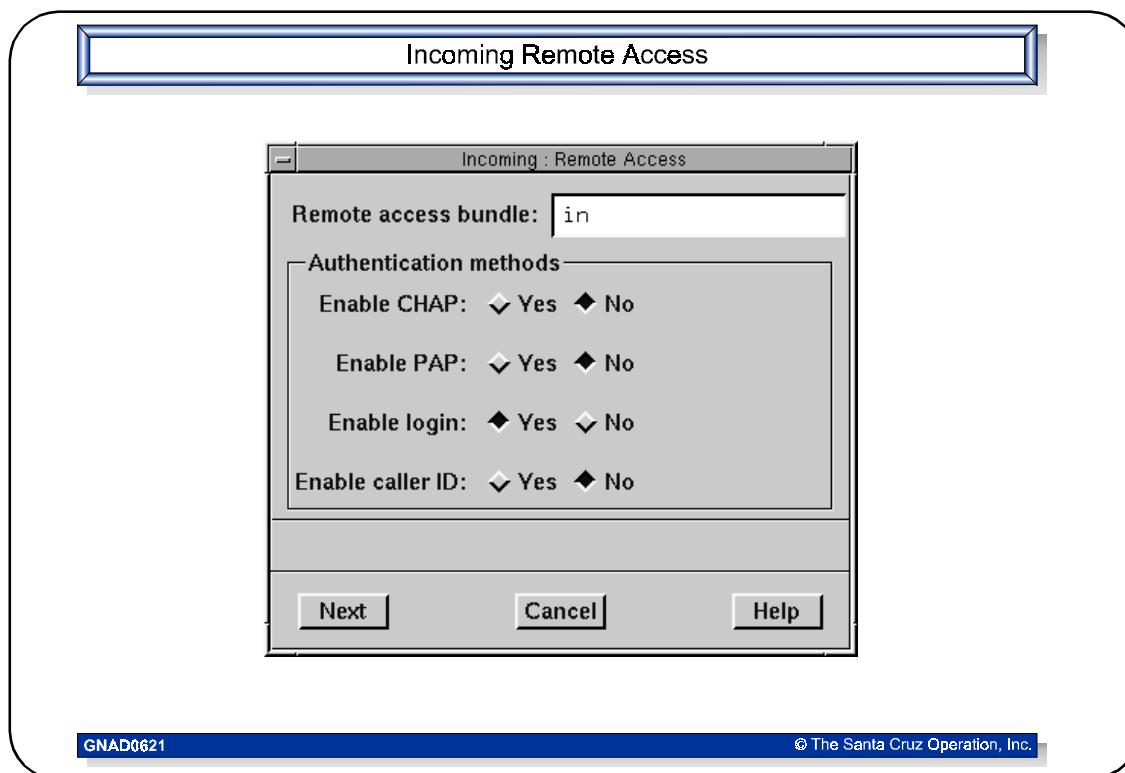
`pppattach bundlename`

where *bundle*name is the name you specified in the “Provider bundle name” field in the **Outgoing Bundle** form.

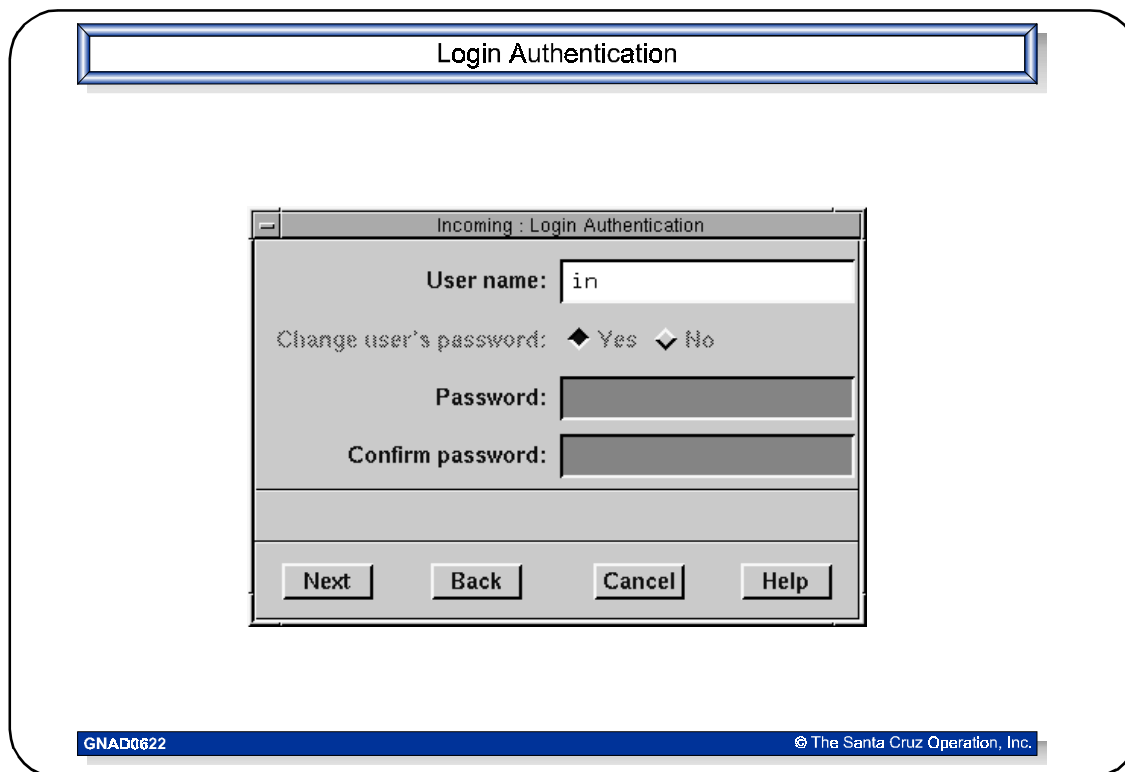
Use the **pppdetach**(1M) command to close a PPP connection. The syntax is:

`pppdetach bundlename`

AV 6-21



AV 6-22

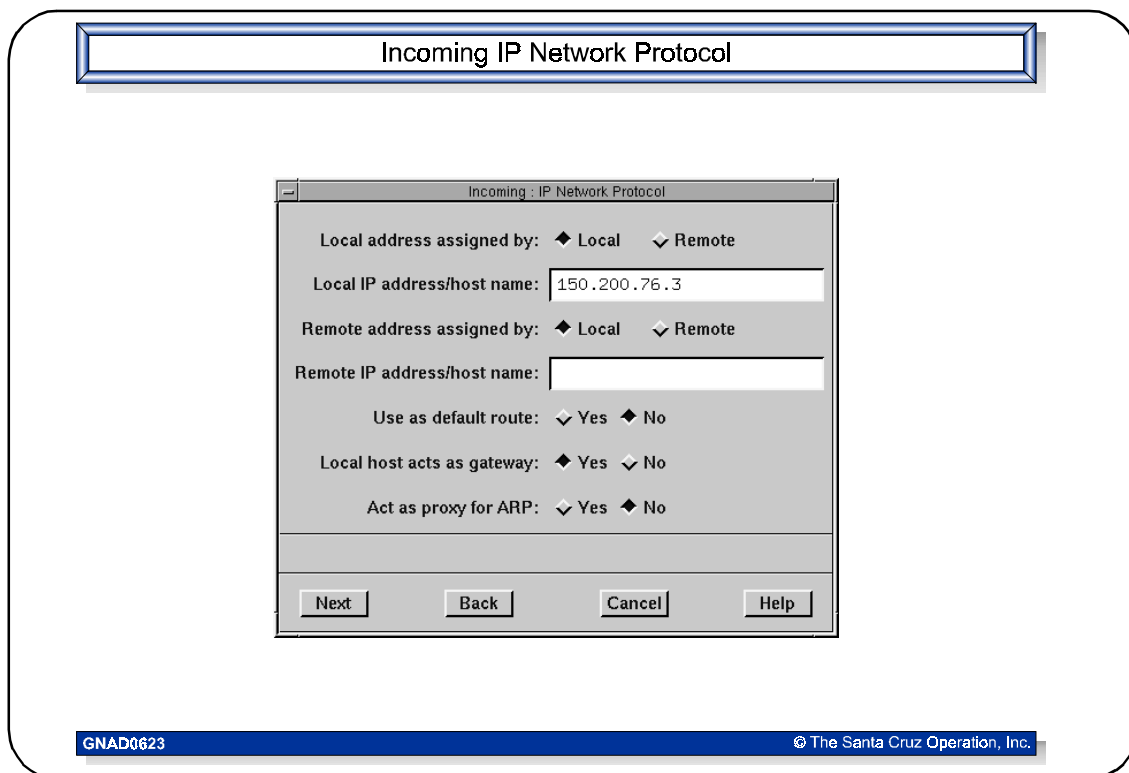


6.3.4 Creating incoming PPP connections

The **Network Configuration Manager** lets you configure both incoming and outgoing PPP connections. Before adding an incoming PPP connection, run the **Modem Manager** and verify that the modem is configured to accept incoming calls.

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select **View** ⇒ **WAN**.
- Step 3. Select **Software** ⇒ **Configure PPP**. This starts the **PPP Manager**.
- Step 4. Select **Wizard** ⇒ **Add**.
- Step 5. Select **Incoming - Remote Access Server** and click on **Start**.
- Step 6. Complete the **Remote Access** configuration form.
 - Remote access bundle
 - Enter a name to identify this connection.
 - Authentication methods
 - Choose one or more authentication method(s),
ChAP, PAP, UNIX login, and caller ID or both.
 - Click on **Next**.
- Step 7. Enter identification and password information for the authentication method(s) selected. Click on **Next** to continue through each authentication form.

AV 6-23



- Step 8. Complete the **IP Network Protocol** configuration form. Enter the TCP/IP parameters related to this PPP link.

Local address assigned by

Specifies whether this machine or the remote machine sets the IP address of the local side of the PPP link.

IP addresses can also be assigned dynamically using the Address Allocation Server (AAS) Refer to **scohelp** ⇒ **Networking** ⇒ **Administering TCP/IP and Internet services** ⇒ **Address Allocation Server (AAS)**.

Local IP address/host name

Defines an IP address or resolvable name to be used for the local side of the PPP link.

If these addresses are being set remotely, you can use sample addresses for these fields which are reset upon link negotiation.

Remote address assigned by

Specifies whether this machine or the remote machine sets the IP address of the remote side of the PPP link.

WARNING: Do not assign the same IP address to different incoming users if it is possible for them to contact the system at the same time.

Remote IP address/host name

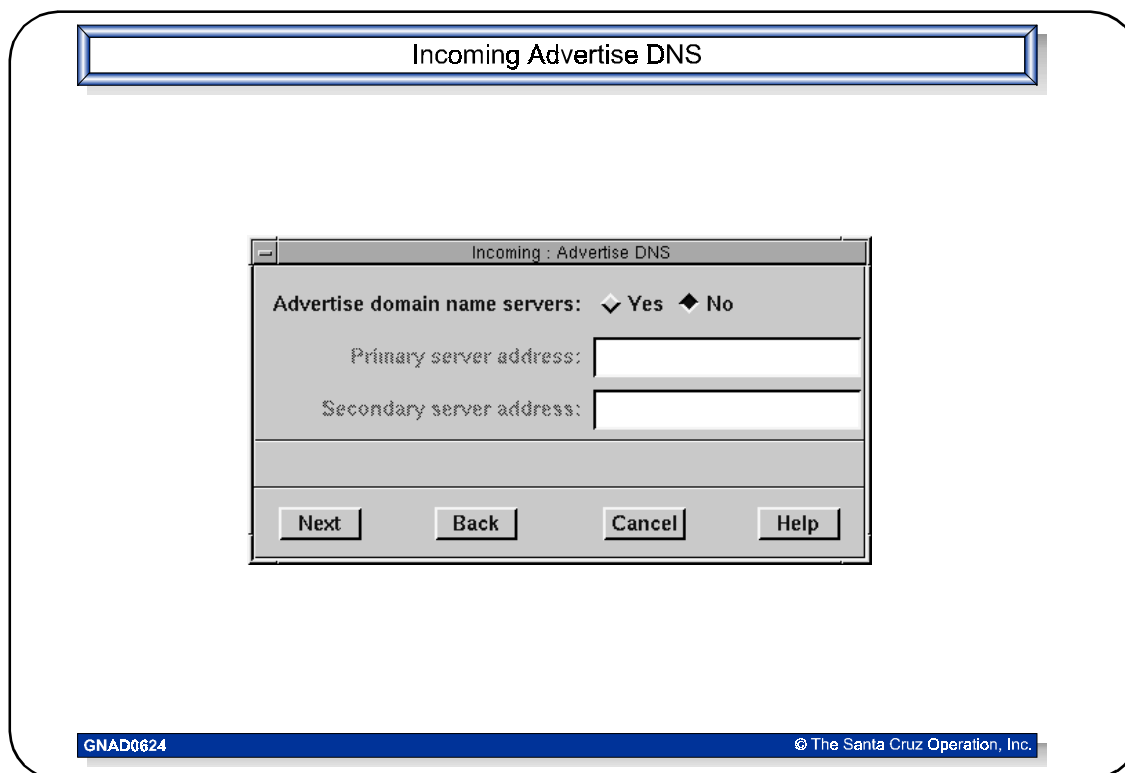
Sets an IP address or resolvable name to be used for the remote side of the PPP link.

- Use as default route A remote access server is typically configured as the gateway to local networks or the Internet. For this reason, this option is usually set to **No**.

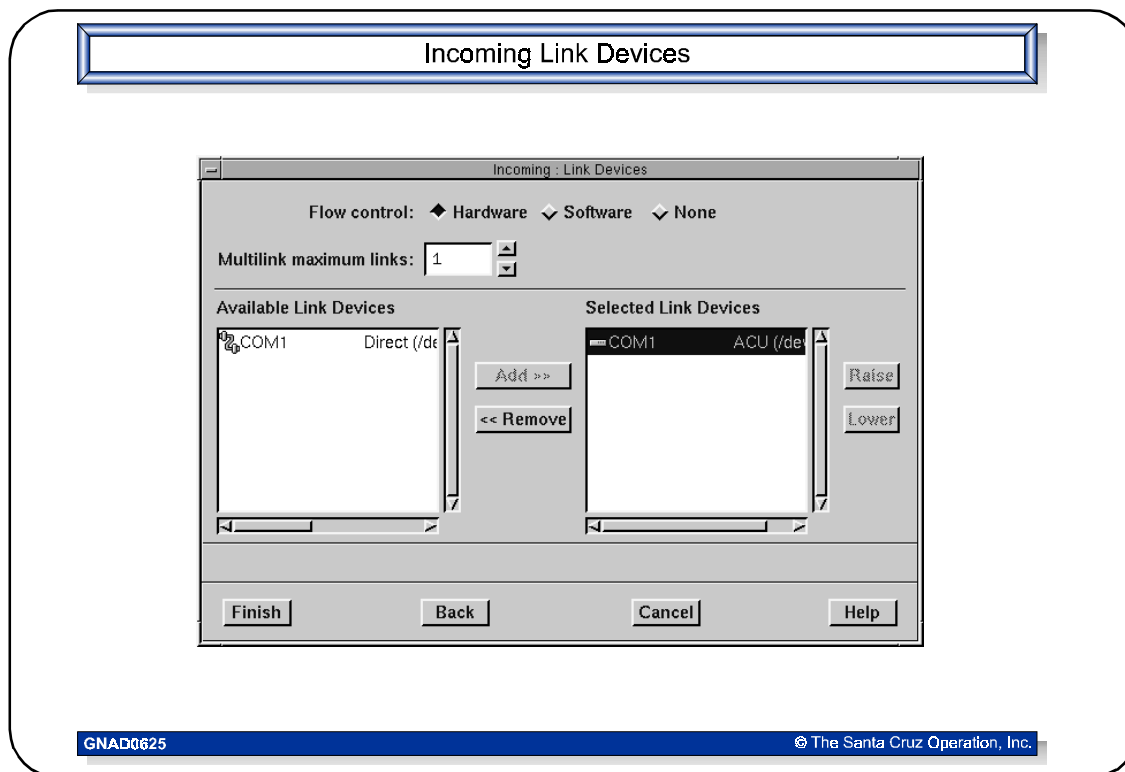
Local host acts as a gateway

Set to **Yes** if this remote access server is the gateway to local networks and the Internet.

AV 6-24



AV 6-25



Act as a proxy for ARP

Use proxy ARP when the host at the other end of the link is using an IP address that belongs to your LAN. If the bundle is being used to connect two distinct networks, or subnets, proxy ARP is not required.

Click on **Next** when you are finished configuring IP parameters.

Step 9. Complete the **Advertise DNS** configuration form.

Advertise DNS name servers

Specifies whether this machine advertises name services to remote systems. The remote system must be configured to use the name servers to resolve host names and IP addresses.

Primary name server Enter the IP address of an alternate server.

Secondary name server

Enter the IP address of an alternate name server.

Click on **Next** when you are finished configuring DNS.

Step 10. Complete the **Link Devices** configuration form to select which devices this bundle can use.

Available Link Devices

Select the device(s) that will be used for the link.

Flow Control

Select the type of flow control to be used between the system and device. Select from **Hardware** (RTS/CTS), **Software** (XON/XOFF), or **None**. Most devices will use **Hardware**. ISDN synchronous-mode devices do not use flow control.

Multiple maximum links

Select the maximum number of links to use in a connection to this host. To disable multilink, choose 1. To enable multilink, choose any number greater than 1.

Multilink must be supported by the remote host. You might also need to configure multiple phone numbers in the **Outgoing Bundle** window if these extra links are not accessible by the same phone number.

Click on **Add** to add the device to the “Selected devices” list. Repeat to add multiple devices.

If you select multiple devices, your connection will be able to use any available device from the group.

- Step 11. Click on **Finish** to complete the incoming PPP connection setup. Click on **OK** to confirm that the configuration completed successfully.
- Step 12. Select **Hardware** ⇒ **Exit** to exit the **Network Configuration Manager**.

6.4 Troubleshooting PPP

Configuring systems to communicate with each other can be a complicated task requiring many components and steps. A problem with any of these components can cause a PPP session to fail so problems with PPP negotiations, especially on new configurations is not rare.

It is important to diagnose which component or parameter is failing to properly configure and repair a connection. Diagnosis of a PPP problem can be attacked systematically through a series of steps:

1. Manually test the connection.
2. Enable and examine PPP logs.
3. Check that the PPP interface exists.
4. Verify network connections between hosts.
5. Check routing.
6. Reset PPP.

6.4.1 Manually testing a connection

Before investigating PPP issues, it is a good idea to verify the underlying communication channel. This is especially important for analog modems and phone lines.

To verify an incoming connection:

- Try calling the line from a normal voice phone on a different line and listening for the distinctive high-pitched sound of a modem answering. This will verify that the phone number is correct and that the modem is set to auto-answer.
- Try calling the system with **cu**(1Mbnu) to verify that the correct prompts are displayed when the phone is answered. The **cu** command is discussed below.

To verify an outgoing connection:

- Use a normal voice phone to verify that there is a dial tone on the phone line.
- Try using the **cu** command to place an outgoing call.

To test the connection between the systems, use these commands:

- **# cu 5551234**

This command causes the modem to call the specified phone number and tries to establish a connection. If this works, your modem and *Devices(4bnu)* file are probably configured correctly.

- **# cu sysname**

This command again tries to contact the remote machine but this time refers to the *Systems(4bnu)* file for the entry corresponding to *sysname* to retrieve connection information. If this fails, check this file for misconfiguration.

- **# cu -x sysname**

By default, **cu(1bnu)** does not use the login script specified in the *Systems(4bnu)* file. The **-x** option enables the use of the script. If the previous commands were successful and this one fails, check the login script.

You may also want to specify:

- **-d** to enable debugging to the screen
- **-l** to specify which device to use

Enter **~ .** to drop a connection after it has been established.

When testing the connection, you might want to enable debugging on the call server daemon. This provides you with more information about the steps the computer took in establishing the connection and any problems it encountered in doing so.

To enable debugging, first you must stop the current **cs(1Mbnu)** daemon:

```
# ps -ef | grep cs
```

```
# kill pid
```

Then, restart it with debugging enabled:

```
# cs -d
```

This causes the **cs(1Mbnu)** daemon to log its activity in */var/adm/log/cs.debug*.

The **cu** command can also be used to send AT commands directly to the modem. Consult your modem hardware documentation for the commands accepted by your modem. The commands used in the examples below work on many, but not all, modems.

To directly connect to the modem on your first serial port, use the command:

cu -l term/00m

If successful, the message **Connected** displays.

After connecting, you can try the following example modem AT commands:

ATE1	Sets the modem to display keystrokes as you type
ATI4	Displays current configuration settings
ATM1	Turns modem speaker on while dialing
ATS0=2	Sets the modem to answer on the second ring
AT&W	Writes current settings to modem's NVRAM
~.	Ends the cu session and disconnects from the modem

6.4.2 Using PPP Logs

If you are able to connect automatically to the remote machine but PPP is still not working correctly, the problem might be with a PPP negotiated parameter. To monitor PPP negotiation, enable debugging on the connection.

The following debug levels are available:

none	Turns off debugging
low	Generates a small amount of output including negotiated values
med	Generates a more detailed output and is usually sufficient for most debugging purposes
high	Is usually only necessary if you want to see all packets and their contents. This can rapidly generate a large quantity of data in the log file.

You can set a debugging level on a link or bundle using the **ppptalk(1M)** command.

To enable **med** level debugging on the bundle called **austin**, enter:

```
# ppptalk debug med bundle austin
```

To enable high level debugging on the link group `link_0`, enter:

```
# ppptalk debug high link link_0
```

Debugging output is sent to the PPP log file. By default, this is `/var/adm/log/ppp.log`.

For more information about the debug levels, see **ppptalk(1M)**.

Help interpreting the PPP log file can be found in **scohelp** ⇒ **Internet and Intranet** ⇒ **Configuring WAN connections** ⇒ **Troubleshooting PPP links** ⇒ **Examine PPP negotiation in the log file**.

6.4.3 Check that the PPP interface exists

If the PPP log file shows that the PPP link has been established, you can then check to make sure the correct network interface has been added to the system. This can be done by:

- Step 1. Use the **ifconfig(1Mtcp)** command with the **-a** option to check if an interface (such as `ppp0`) for the link exists.
- Step 2. Verify that the displayed flags indicated that it is marked UP and that the IP addresses for the ends of the link are what you expect.

 If an interface does not exist for the PPP link, the link might have been taken down because the line was too noisy.
- Step 3. Re-examine the PPP log file to discover where a problem might be occurring.

6.4.4 Verify network connection between hosts

If the interface appears to be up, verify network connectivity between the local and remote host. You can do this with any network command, such as **telnet**, but **ping(1Mtcp)** is tailor-made for this purpose:

```
# ping -ns remote_IP_address
```

- If **ping** fails, check the network interface again to see if it has gone down.
 - If it is down, check the PPP log file to find out why.
 - If it is up and you cannot get network connectivity between the machines, the problem is likely a TCP/IP configuration problem. Refer to **scohelp** ⇒ **Networking** ⇒ **Administering TCP/IP and Internet services** ⇒ **Troubleshooting TCP/IP**.
- If **ping** succeeds, you have a working connection between the tested machines. You should test several systems, including some across routers.

6.4.5 Check routing

If you can connect to the remote machine but not to the rest of its network, there is a routing problem. These problems can include:

- The remote system is not forwarding packets. Verify that the remote system is configured as a router as described in Section 5.1, “Configuring a System as a Router,” on page 5-5.
- The remote machine is not doing proxy ARP. Check the configuration of this interface on the remote machine. Running **arp -a** on the remote system should show the proxied ARP addresses.
- The routing tables might not be correct. Check the routing tables on both sides using **netstat -rn**. If the routes are incorrect, they can be fixed with the **route(1Mtcp)** command. For example, to set up a default route:

```
route add default remote_IP_address
```

For more information, refer to **scohelp** ⇒ **Networking** ⇒ **Administering TCP/IP and Internet services** ⇒ **Setting up subnets and routers** ⇒ **Troubleshooting IP routing**.

6.4.6 Reset PPP

If all else fails, try resetting a bundle or a link group or even restarting the PPP daemon.

When you reset a bundle or a link, existing connections are destroyed and the configuration files are updated.

To reset a bundle called **austin**, enter:

```
# ppptalk reset bundle austin
```

To reset a link group called **link_0**, enter:

```
# ppptalk reset link link_0
```

To stop the ppp daemon, enter:

```
# ppptalk stop
```

To restart the ppp daemon, enter:

```
# /usr/sbin/pppd
```

6.5 Lesson Summary

This lesson discussed the WAN network interfaces types:

- ISDN
- Frame relay
- Leased Lines
- Cable modems
- ADSL

It also discussed the WAN interface protocols:

- PPP
- SLIP

A thorough discussion of the PPP protocol including setting up incoming and outgoing connections was also provided.

Review questions

1. Which WAN technology would you use to implement a corporate intranet consisting of multiple geographically disperse satellite offices connecting to a central office?
2. Which files contain information about the configured modems available for WAN connections? Which ones specify the phone numbers and login scripts used to contact remote systems.
3. In which situations would you enable proxy ARP on a bundle? In which situations would you not enable proxy ARP on a bundle?

6.6 Lab: Configuring PPP Connections

The goal of this lab is to configure incoming and outgoing PPP connections using the **PPP Manager**.

You will need two computers for this lab, one for the outgoing connection and one for the incoming connection.

Preparation

Before beginning this lab, your instructor will supply you with this information:

- The phone number for the instructor's dial-in system.
- The username and password used for PPP dial-in access on the instructor's dial-in system.
- The IP address of a name server accessible from the instructor's dial-in system.

6.6.1 Disabling LAN Access

Before configuring an outgoing PPP connection, you need to disable LAN access from the system that will have the outgoing connection. If you left your system connected to the LAN, it might become difficult to test your PPP configuration, as you would not know whether the PPP interface or the LAN interface was used to contact a remote system.

To disable LAN access, enter the following commands:

```
inconfig net0 down  
route flush
```

After the lab, reboot your system to re-enable your network interface and re-initialize your routing table.

6.6.2 Configuring a Modem for Outgoing Calls

If you have not already done so, configure a modem on the system that will be used for the outgoing connection.

Procedure

- Step 1. Run **SCOadmin**.
- Step 2. Select **Networking** ⇒ **Network Configuration Manager**.
- Step 3. Select **View** ⇒ **WAN**.
- Step 4. Select **Hardware** ⇒ **Add new WAN Device** ⇒ **Modem**. This starts the **Modem Manager**. Modems already configured on the system are displayed.
- Step 5. Select **Modem** ⇒ **Add**.
- Step 6. Select either **Automatic Detection** or **Manual Configuration**.

If you select **Automatic Detection**:

- Select which ports you want to scan from the popup window.
- Click on **Detect**.
- Click on **OK** to verify that your modem is on.

If automatic detection fails, try manual configuration.

Select **Modem** ⇒ **Add** to return to the **Manual Configuration** option.

If you select **Manual Configuration**:

- Select the “Modem vendor” and “Modem model” from the lists.
- Select the port the modem uses in the “Modem port” field.
- Click on **OK**.

- Step 7. Modify the serial port configuration, if necessary. Select the modem you added in Step 6. Select **Modem** ⇒ **Modify**. Click on **Configure port** to start the **Serial Manager**.

When finished, click on **OK** in the **Modify Serial Port** window.

- Step 8. Click on **OK** in the **Modem Configuration** window.

- Step 9. Select **Host** ⇒ **Exit**.

- Step 10. Test your modem with the command:

cu *phonenumber*

where *phonenumber* is the number your instructor gave you.

Press  to see the login prompt.

6.6.3 Configuring an Outgoing PPP Connection

Configure your system to call another machine to initiate a PPP connection using the **PPP Manager**.

In this part of the lab, you will configure your system to call a system previously configured by your instructor. This will allow you to test your outgoing PPP connection by itself. Later in the lab you will configure an incoming system as well.

Procedure

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select **View** ⇒ **WAN**.
- Step 3. Select **Software** ⇒ **Configure PPP**. This starts the **PPP Manager**.
- Step 4. Select **Wizard** ⇒ **Add** to add a new connection.
- Step 5. Select **Outgoing-Internet/Intranet**. Click on **Start**.
- Step 6. Complete the **Bundle** configuration form, including:
- | | |
|----------------------|---|
| Provider bundle name | Choose a name that will be used to bring up this link with pppattach(1M) . |
| Remote system | Choose a name used in the <code>/etc/uucp/Systems</code> file to refer to this connection. This does not need to match the actual name of the remote system. It can be the same as the bundle name. |
| Telephone numbers | Enter the phone number your instructor gave you and click on Add . |
| How to connect | Select Use pppattach . |
- Click on **Next** when you are finished configuring the provider details.
- Step 7. Select **None** for authentication protocol. Click on **Next**.

- Step 8. Complete the **Login Script** configuration form.
- In the “2nd Send” field, enter the user name your instructor gave you.
- In the “3rd Send” field, enter the password your instructor gave you.
- Click on **Next** when you are finished configuring authentication.
- Step 9. Complete the **IP Network Protocol** configuration form. Enter these TCP/IP parameters:
- Local address assigned by
Select **Remote**.
- Local IP address/host name
Leave blank.
- Remote address assigned by
Select **Remote**.
- Remote IP address/host name
Leave blank.
- Use as a default route
Select **Yes**.
- Local host acts as a gateway
Select **No**.
- Act as a proxy for ARP
Use the setting provided by your instructor.
- Click on **Next** when you are finished configuring IP parameters.

- Step 10. Complete the **Name Servers** configuration form, including:
- Local domain Specify the fully-qualified domain name for the classroom network.
 - Name server search order
Enter IP address(es) of the classroom name server(s) provided by your instructor. Click on **Add** for each name server entered.
- Click on **Next** when you are finished configuring DNS.
- Step 11. Complete the **Link Devices** configuration form:
- Flow Control Select **Hardware** flow control.
 - Multilink maximum links
Select **1**.
 - Available Link Devices
Select your modem. Click on **Add**.
- Step 12. Click on **Finish** to complete the outgoing PPP connection setup.
- Step 13. Click on **OK** to confirm that the configuration completed successfully.
- Step 14. Select **Host** ⇒ **Exit** to exit the **PPP Manager**. The UUCP and PPP configuration files shown in AV 6-10 are now updated.
- Step 15. Initiate your connection with the command:
- pppattach *providername***
- where *providername* is the name you entered in Step 6.
- Step 16. Test your connection by using **ping**, **telnet**, and other networking commands to access systems on the remote side of the link.

6.6.4 Configuring a Modem for Incoming Calls

If you have not already done so, configure a modem on the system that will be used for the incoming connection.

Procedure

- Step 1. Start **SCOadmin** on the system that will be used for the incoming connection.
- Step 2. Select **Networking** ⇒ **Network Configuration Manager**.
- Step 3. Select **View** ⇒ **WAN**.
- Step 4. Select **Hardware** ⇒ **Add new WAN Device** ⇒ **Modem**. This starts the **Modem Manager**.
- Step 5. Select **Modem** ⇒ **Add** ⇒ **Manual Configuration**.
- Step 6. Select your “Modem vendor” and “Modem model” from the lists.
- Step 7. Select the port the modem is using in the “Modem port” field.
- Step 8. Click on **Configure port**.
- Step 9. Select **incoming**.
- Step 10. Select a “Speed” of 57600 bps.
- Step 11. Click on **OK** twice.
- Step 12. Select **Host** ⇒ **Exit**.
- Step 13. Test your modem with the command:

cu *phonenumber*

where *phonenumber* is the number your instructor gave you.

If you see a login prompt, the test was successful. You may have to press  to see a login prompt.

6.6.5 Configuring an Incoming PPP Connection

Using the **PPP Manager**, configure a system as a remote access server which will answer incoming PPP calls. Do not use the same system you used in the outgoing connection lab.

Procedure

- Step 1. Start the **Network Configuration Manager**.
- Step 2. Select **View** ⇒ **WAN**.
- Step 3. Select **Software** ⇒ **Configure PPP**. This starts the **PPP Manager**.
- Step 4. Select **Wizard** ⇒ **Add**.
- Step 5. Select **Incoming - Remote Access Server** and click on **Start**.
- Step 6. Complete the **Remote Access** configuration form.
- Remote access bundle
Enter **dialin1**.
- Authentication methods
Choose **Enable login**.
- Click on **Next**.
- Step 7. Enter identification and password information for the UNIX account:
- User name **ppptest**
- Password **Safe1!**
- Click on **Next**.
- Step 8. Complete the **IP Network Protocol** configuration form. Enter the TCP/IP parameters related to this PPP link.
- Local address assigned by
Select **Local**.
- Local IP address/host name
Leave your IP address in this field.

Remote address assigned by
Select **Local**.

Remote IP address/host name
Enter an unused IP address on your local network.

Use as default route Select **No**.

Local host acts as a gateway
Select **Yes**.

Act as a proxy for ARP
Select **Yes**.

Click on **Next** when you are finished configuring IP parameters.

Step 9. Select **No** in the **Advertise DNS** configuration form. Click on **Next**.

Step 10. Complete the **Link Devices** configuration form:

Flow Control Select **Hardware** flow control.

Multilink maximum links
Select **1**.

Available Link Devices
Select your modem. Click on **Add**.

Step 11. Click on **Finish** to complete the outgoing PPP connection setup.

Step 12. Click on **OK** to confirm that the configuration completed successfully.

Step 13. Select **Hardware** ⇒ **Exit** to exit the **Network Configuration Manager**.

6.6.6 Configuring an Outgoing Connection Revisited

Add a second outgoing connection to the system you used in Section 6.6.3, “Configuring an Outgoing PPP Connection,” on page 6-59.

This time, configure an outgoing connection to the remote access server that you just finished configuring.

Procedure

Repeat the steps from Section 6.6.3, “Configuring an Outgoing PPP Connection,” on page 6-59, with the following parameters:

- Use the name of your remote access server as the “Provider name” and “Remote system name” in Steps 6.
- Use the phone number of your remote access server system in Step 6.
- Use the user name `ppptest` and the password `Safe1!` in Step 8.
- Enter the address of a name server reachable from your incoming connection system in Step 10.

6.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

arp (1Mtcp)	Address resolution display and control
cu (1bnu)	Connects to a configured device or system.
cs (1bnu)	Connection Server, a daemon that establishes connections for TLI/serial network services
netcfg (1M)	Network configuration utility which can be invoked from the command line, or via scoadmin (1M)
ping (1Mtcp)	Tests network connections by eliciting hosts to reply to packets sent to them
pppattach (1M)	Brings up <i>manual</i> or <i>on demand</i> outgoing PPP connections
pppdetach (1M)	Drops (disconnects) PPP connections
pppd (1M)	The PPP daemon
ppptalk (1M)	Administrative interface to the PPP subsystem. Allows configuration of the PPP system and communication with the PPP daemon
route (1Mtcp)	Allows manually manipulation of the routing tables
scoadmin (1M)	Menu-driver program for configuring the UnixWare 7 system. It includes the Network Configuration Manager .

Relevant files

<i>/etc/uucp/Devices</i>	Contains one entry per device to be used by UUCP. Refer to the <i>Devices(4bnu)</i> manual page.
<i>/etc/uucp/Systems</i>	Contains one entry per system connected via UUCP. Refer to the <i>Systems(4bnu)</i> manual page.
<i>/etc/ppp.d/.pppcfg</i>	Contains the PPP configuration information, including bundle definitions. Refer to the ppptalk (1M) manual page.

6.8 Answers to Review Questions

1. Frame Relay would likely be the best choice for this scenario. Using Frame Relay would avoid the tolls on long distance connections for the satellite offices and would require less equipment and lines into the central office.
2. The `/etc/uucp/Devices` file contains information about available modems, their type, and to which serial port they are connected. The `/etc/uucp/Systems` file contains information about remote systems such as their phone numbers and the login scripts necessary to establish a connection with them.
3. Use proxy ARP when the host at the other side of the link is using an IP address that belongs to your LAN. If the bundle is being used to connect two distinct networks, or subnets, proxy ARP is not required as normal routing protocols handle this situation.

7

TCP/IP Administration Topics

AV 7-1

The slide is titled "TCP/IP Administration Topics" in a blue-bordered box at the top. Below the title, there is a bulleted list of topics. At the bottom of the slide, there is a blue footer bar containing the text "GNAD0701" on the left and "© The Santa Cruz Operation, Inc." on the right.

- TCP/IP startup
- Daemon startup
- The **inetd**(1Mtcp) daemon
- the */etc/inet/config* file
- Testing daemons
- Configuring trusted access

GNAD0701 © The Santa Cruz Operation, Inc.

Lesson 7

TCP/IP Administration Topics

Introduction

Previous lessons have described how to design and build a LAN and WAN using facilities available in UnixWare 7. There are additional tasks you must perform to maintain a TCP/IP network.

This lesson discusses the administrative tasks of starting, stopping, and configuring TCP/IP functionality. After you understand the issues and can perform the functions presented in this lesson, you will be able to perform your job as a network administrator with greater ease.

Objectives

This lesson covers TCP/IP administration issues. By the end of this lesson, you will be able to:

- Describe how TCP/IP is started in UnixWare 7
- Configure how daemons are started
- Configure services started by the **inetd**(1Mtcp) daemon and how they are started
- Configure services started by */etc/inet/config*
- Use **telnet**(1M) to test daemons
- Configure and explain the functionality of trusted user access

7.1 TCP/IP Start-up

TCP/IP is started when your system boots into multiuser mode and stopped when you shut down into single-user mode. Note that rebooting is the only way to restart TCP/IP in UnixWare 7. Understanding the mechanics of TCP/IP startup and shutdown will help you troubleshoot network-related problems that can occur.

TCP/IP start-up involves the following:

- Initializing protocol-independent streams modules
- Initializing the IP, TCP, and UDP streams modules with **slink**(1Mtcp)
- Initializing the network interfaces with **ifconfig**(1Mtcp)
- Starting server daemons
- Installing packet filtering streams modules
- Initializing PPP, SNMP, and RPC

UnixWare 7 uses several Bourne-shell and TCL scripts, binary programs, and configuration files to accomplish these tasks.

7.1.1 Configuring protocol-independent STREAMS modules

The SCO operating systems, including UnixWare 7, are designed to support multiple networking protocols, including TCP/IP. To facilitate this support, the protocol stacks are built using STREAMS modules.

The protocol-independent streams modules are loaded and initialized by the `/etc/rc2.d/S15nd` script, which is hard-linked to the `/etc/init.d/nd` script. This script:

- Is part Bourne-shell and part Vtcl
- Invokes the **ndcfg**(1M) program to configure the MDI network adapter drivers, and then the DLPI streams module above those drivers

AV 7-2

Example ifconfig(1Mtcp) Commands**Configuring Interfaces:**

```
# /etc/ifconfig lo0 127.0.0.1 perf 57344 57344 1
# /etc/ifconfig -p net0 200.100.50.1 netmask 255.255.255.0 \
> broadcast 200.100.50.255 perf 24576 24576 1
```

Listing Interfaces:

```
# /etc/ifconfig -a
net0: flags=4043<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 200.100.50.1 netmask ffffffff broadcast 200.100.50.255
    perf. params: recv size: 24576; send size: 24576; full-size frames: 1
    ether 00:00:c0:83:47:8f
lo0: flags=4049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 8232
    inet 127.0.0.1 netmask ff000000
    perf. params: recv size: 57344; send size: 57344; full-size frames: 1
```

Bringing interfaces down and back up:

```
# /etc/ifconfig net0 down
# /etc/ifconfig net0 up
```

GNAD0702

© The Santa Cruz Operation, Inc.

7.1.2 Configuring basic Internet services

Basic Internet services are configured from the */etc/init.d/inetinit* file. This Bourne-shell script:

- Is hard-linked to */etc/rc2.d/S69inet* for automatic run at system startup
- Is hard-linked to */etc/rc0.d/K69inet* and */etc/rc1.d/K69inet* for automatic run at system shutdown
- Initializes the IP, TCP, and UDP streams modules by invoking the **slink**(1Mtcp) command using the directives found in the */etc/inet/strcf* file.
- Invokes the */usr/sbin/initialize* program, which reads the */etc/confnet.d/inet/interface* file, and configures each interface by:
 - Running **slink**(1Mtcp)
 - Running **ifconfig**(1Mtcp) with the other parameters in the entry to set the interface's IP address, netmask, broadcast address, etc.

The **ifconfig** command can also be run from the command line. AV 7-2 illustrates how to use the **ifconfig** command.

- Starts basic Internet support daemons as indicated by the */etc/inet/config* file. The format of this file is documented in the *config(4tcp)* manual page.

Exercise: Where does ifconfig(1Mtcp) get its information

Compare the contents of the */etc/confnet.d/inet/interface* file with the results of the **ifconfig -a** command. Answer these questions:

Is the number of interfaces listed by **ifconfig** the same as the number of entries in the interface file?

Do the IP addresses, netmasks, and broadcast addresses in the interface file show up in the output of **ifconfig**?

7.1.3 Installing packet filtering

Packet filters allow network administrators to limit which packets they are willing to transmit from and receive into their systems. Packet filters are:

- One brick you can use to build a firewall
- Implemented in the DLPI streams module and are installed on all configured interfaces via `/etc/rc2.d/S70pf` file

See Section 10.2.1, “Packet filters,” on page 10-13 for further information on how this script works.

7.1.4 Initializing PPP, SNMP, and RPC

The next step in starting up TCP/IP on UnixWare 7 is the starting of support for the PPP, SNMP, and RPC protocols. This is accomplished as follows:

PPP Is started by the `/etc/rc2.d/S71ppp` script. What this script does to start PPP, what daemons are used to support PPP, and other related issues are discussed in section Section 6.2.2, “Point-to-Point Protocol (PPP),” on page 6-21.

SNMP The Simple Network Management Protocol daemons are started and stopped by the `/etc/init.d/snmp` file. This file:

- Is a Bourne-shell script
- Is linked to `/etc/rc2.d/S73snmp` so that it is started automatically at system startup
- Starts up the **snmpd**(1Msnmp) and **hostmibd** programs

Detailed discussion of SNMP is beyond the scope of this course. See **scohelp**(1X) for more information on SNMP.

RPC The Remote Procedure Call (RPC) services are started by the `/etc/rc2.d/S75rpc` file. RPC is needed for NFS and other networking services.

7.2 Daemon Startup

This course has discussed how to set up the TCP/IP infrastructure, including how to:

- Install and configure affiliated hardware
- Configure the TCP/IP protocol stack

This infrastructure carries, routes, and delivers data between client and server programs running on machines on your intranet. Without the client and server programs, there would be no need for the infrastructure.

The client programs are normally started by users whenever they want to accomplish a task (e.g., transfer a file or log into a remote system). The server programs are implemented as daemon programs and are started in one of four ways:

1. Through an entry in an system startup script. A daemon started this way is said to be a *standalone* daemon because it does not need any other program to start it up or watch over it.
2. Through an entry in the `/etc/inet/config` file. As discussed in Section 7.1.2, “Configuring basic Internet services,” on page 7-7, this file is read by the `/usr/sbin/initialize` program at boot time. Daemons started this way are also said to be standalone. Configuration of this file is discussed in Section 7.2.1, “The config(4tcp) file,” on page 7-13.
3. Through an entry in the `/etc/inet/inetd.conf` file, in which case the daemon is actually started by the **inetd**(1Mtcp) daemon each time it is needed. Configuration of this file is discussed in Section 7.3, “The inetd(1Mtcp) Super Daemon,” on page 7-17.
4. Through the Service Access Facility (SAF). This method, though possible, is rarely used and is therefore not discussed in this course.

You can decide how to start many of the daemons. There are trade-offs to be considered when deciding whether a daemon should be started standalone or by **inetd**. The arguments against a standalone daemon include:

- If a service is rarely used, it is wasting system resources by running all the time.
- If a service is heavily used, then one service daemon might not be enough to keep up with the demand.

The arguments in favor of starting a daemon standalone include:

- There is no latency associated with **inetd** having to start the daemon with each new connection.
- Each new connection causes a new daemon to be started. If there are several connections being made, and the daemon is large, many resources are consumed by the multiple copies of the daemon.

Standalone daemons have their own script if they need to be terminated in a special way during system shutdown. Otherwise, they can be started by entries in the */etc/inet/config* file.

These, and other issues were considered when SCO decided which daemons should be started from **inetd**, SAF, scripts, and *initialize*. For example:

- The **lpd** services are started by SAF.
- Routing services (by either **routed**(1Mtcp) or **gated**(1Mtcp)) are started from entries in the */etc/inet/config* file.
- The **httpd** services for **scohelp**(1X) are started standalone by the */etc/rc2.d/S93scohelphttp* file.
- The **in.rlogind**(1Mtcp) is started by **inetd**(1Mtcp).

AV 7-3

The /etc/inet/config File

```

1:/usr/eac/bin/initsock::y:/usr/eac/bin/initsock::
1a:/usr/sbin/inconfig::y:/etc/inet/inet.dfl::
2:/usr/sbin/syslogd::y:/etc/syslog.conf::
5:/usr/sbin/in.named::y:/etc/inet/named.boot::
3:/usr/sbin/in.pppd::y:/etc/inet/ppphosts::
4a:/usr/sbin/in.gated::Y:/etc/inet/gated.conf::
# The next line can be read:
# If field 3 is a Y and /usr/sbin/in.gated was not run, then run
# /usr/sbin/in.routed
4b:/usr/sbin/in.routed:/usr/sbin/in.gated:Y::
#4c:/usr/sbin/route::n::add default router_placeholder:
6:/usr/sbin/in.xntpd::y:/etc/inet/ntp.conf::
7:/usr/bin/sh::y:/etc/inet/rc.inet:/etc/inet/rc.inet start:
# By default start address allocation and dhcp servers
8a:/usr/sbin/in.aasd::y:/etc/inet/aasd.conf::
#8b:/usr/sbin/in.dhcpd::y:/etc/inet/dhcpd.conf:-t 0:
# Entry to start bootpd, comment-out dhcpd line
#9a:/usr/sbin/in.bootpd::y:/etc/inet/bootptab:-t 0:
# Entries to start both bootpd and dhcpd, comment out dhcpd entry
#9b:/usr/sbin/in.bootpd::y:/etc/inet/bootptab:-t 0 -S -P bootps-alt:
#9c:/usr/sbin/in.dhcpd::y:/etc/inet/dhcpd.conf:-t 0 -b bootps-alt:
4c:/usr/sbin/route::y::add default 198.140.193.1:
4d:/usr/sbin/route::y::add 224.0.0.0 198.140.193.111:

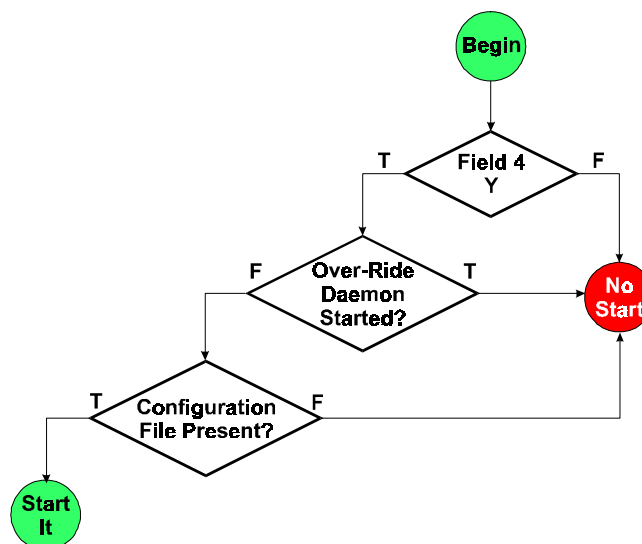
```

GNAD0703

© The Santa Cruz Operation, Inc.

AV 7-4

The /etc/inet/config Flow Chart



GNAD0704

© The Santa Cruz Operation, Inc.

7.2.1 The *config(4tcp)* file

To understand how daemons are started by the */usr/sbin/initialize* program from the */etc/inet/config* file, you need to understand the format of the file. It is documented in the *config(4tcp)* manual page.

As AV 7-3 illustrates, this file contains 6 fields, each separated by a colon. The format of the file is:

```
sort-key:daemon:overriding-daemon:flag:config-filename:options:
```

AV 7-4 illustrates when a daemon is started from the *config* file. For example, examine the lines that read:

```
4a:/usr/sbin/in.gated::Y:/etc/inet/gated.conf::  
4b:/usr/sbin/in.routed:/usr/sbin/in.gated:Y:::
```

The first line starts the */usr/sbin/in.gated* daemon if and only if the */etc/inet/gated.conf* file exists.

The second line indicates that */usr/sbin/in.routed* should be started if and only if the entry in the previous line (*/usr/sbin/in.gated*) did not start because either:

- The 4th field contains an N.
- The specified executable does not exist.
- The specified configuration file does not exist. NOTE: The */etc/inet/gated.conf* file does not exist by default; therefore, the **routed** daemon starts by default.

The */etc/inet/config* file can be edited by hand. Changes made to this file take effect the next time TCP/IP started.

7.2.2 Running programs from */etc/inet/config*

These daemons are available on your system. The programs are started from the */etc/inet/config* file:

initsock	This program provides backwards compatibility to TCP/IP based programs from OpenServer Release 5. This program is located in <i>/usr/eac/bin/initsock</i> .
inconfig (1Mtcp)	This program, when run with no arguments, reads the <i>/etc/inet/inet.dfl</i> file to set initial networking related kernel parameters.
syslogd (1Mtcp)	This program logs system messages to the console and to a file, typically <i>/var/adm/syslog</i>. For more information, see the syslogd(1Mtcp) manual page. It is only started if the <i>/etc/syslog.conf</i> control file is present. The control file is documented by the <i>syslog.conf</i>(4bsd) manual page.
named (1Mtcp)	The named daemon will only be started if you have configured an <i>/etc/inet/named.conf</i> file. For further information about named , see Lesson 8: “Domain Name System (DNS) Configuration.”
in.pppd	This entry does nothing as there is not a <i>/usr/sbin/in.pppd</i> file. The PPP daemon is started by the <i>/etc/rc2.d/S71ppp</i> script.
routed (1Mtcp) or gated (1Mtcp)	You can choose one of these two daemons to maintain your routing information. You must configure the file <i>/etc/inet/gated.conf</i> for gated to start.
xntpd (1Mtcp)	The Internet time daemon will only start if you have created an <i>/etc/inet/ntp.conf</i> file. This daemon is used to synchronize the kernel’s clock using time servers on the Internet. NTP is discussed further in Section 9.7, “Configuring Time Synchronization,” on page 9-63.
<i>rc.inet</i>	This entry is for backwards compatibility. The <i>/etc/inet/rc.inet</i> script currently does nothing.

aasd(1Mtcp) The address allocation server daemon (**aasd**) will only start if the `/etc/inet/aasd.conf` file is present. By default, this file is not present.

For further information on the address allocation server, see the **aasd(1Mtcp)** manual page and Section 11.4.1, “Configuring an Address Allocation Server,” on page 11-55.

dhcpcd(1Mtcp) The dynamic host configuration protocol daemon (**dhcpcd**) is discussed in Section 11.4.2, “Configuring a DHCP server,” on page 11-59.

bootpd(1Mtcp) The boot protocol daemon (**bootpd**) is discussed in the **bootpd(1Mtcp)** manual page.

NOTE: The **dhcpcd** and **bootpd** entries should not be uncommented by hand. They will be uncommented by the **DHCP Manager** as necessary. See Section 11.4.2, “Configuring a DHCP server,” on page 11-59.

route(1Mtcp) Routes are added at the physical end of the file. However their sort key makes them run earlier. The last two physical entries add a default route for the unicast and multicast networks.

The **route** command was discussed in Section 5.2.5, “Manual updates and route(1Mtcp),” on page 5-19.

Some of the daemons started through script files were discussed in Section 7.1.4, “Initializing PPP, SNMP, and RPC,” on page 7-9. Other daemons started through script files are discussed in the lessons covering those services.



The /etc/inet/inetd.conf File

```
# <service_name> <socket_type> <proto> <flags> <user> <server_pathname> <args>
#
# Ftp and telnet are standard Internet services.
#
ftp      stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.ftpd -a
#ftp     stream  tcp      nowait  root    /usr/sbin/in.ftpd    in.ftpd -a
telnet   stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.telnetd
#telnet  stream  tcp      nowait  root    /usr/sbin/in.telnetd  in.telnetd
#
# Tnamed serves the obsolete IEN-116 name server protocol.
#
name      dgram    udp      wait     root    /usr/sbin/in.tcpd    in.tnamed
#name     dgram    udp      wait     root    /usr/sbin/in.tnamed  in.tnamed
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
shell     stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rshd
#shell    stream  tcp      nowait  root    /usr/sbin/in.rshd    in.rshd
login     stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rlogind
#login    stream  tcp      nowait  root    /usr/sbin/in.rlogind  in.rlogind
exec      stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rexecd
#exec     stream  tcp      nowait  root    /usr/sbin/in.rexecd  in.rexecd
comsat    dgram    udp      wait     root    /usr/sbin/in.tcpd    in.comsat
#comsat   dgram    udp      wait     root    /usr/sbin/in.comsat  in.comsat
talk      dgram    udp      wait     root    /usr/sbin/in.tcpd    in.otalkd
#talk     dgram    udp      wait     root    /usr/sbin/in.otalkd  in.otalkd
ntalk     dgram    udp      wait     root    /usr/sbin/in.tcpd    in.talkd
#ntalk    dgram    udp      wait     root    /usr/sbin/in.talkd   in.talkd
pop-3     stream  tcp      nowait  root    /usr/sbin/in.tcpd    /etc/popper
```

GNAD0705

© The Santa Cruz Operation, Inc.

7.3 The *inetd(1Mtcp)* Super Daemon

The **inetd** daemon listens for connection requests for daemons that provide Internet services (e.g., **rlogind** and **telnetd**), and then spawns the requested server process. The daemons that **inetd** listens for are listed in the */etc/inet/inetd.conf* file by default, though this can be changed when **inetd** is started.

The format of the */etc/inet/inetd.conf* file is provided in the *inetd.conf(4tcp)* manual page. AV 7-5 shows a portion of a sample *inetd.conf* file. This file:

- Lists all the server daemons that should be started by **inetd** (e.g., **rlogind** and **telnetd**)
- Has several entries commented out, preventing **inetd** from starting them
- Has several entries commented out that provide alternative ways for **inetd** to run daemons
- Can be modified using a standard text editor and the procedure is outlined in Section 7.3.1, “Configuring *inetd* through the *inetd.conf* file,” on page 7-18.

7.3.1 Configuring `inetd` through the `inetd.conf` file

To change the options for a server daemon that is started by **inetd**, edit the `/etc/inet/inetd.conf` file, then force **inetd** to re-read the file.

In UnixWare 7, **inetd** is run from the Service Access Facility (SAF). To have **inetd** re-read its configuration file, you can either:

- Signal the process running **inetd** with the hang-up (HUP) signal (signal 1)
- Stop and restart the port monitor that is running **inetd**

The following exercise demonstrates how you can modify the `/etc/inet/inetd.conf` file and causes **inetd**(1Mtcp) to re-read the file using the HUP signal.

Exercise: Sending `inetd` the HUP signal

- | | |
|---------|---|
| Step 1. | Verify that you can telnet to your local host. Enter:

telnet localhost |
| Step 2. | Edit the <code>/etc/inet/inetd.conf</code> file and enter a pound sign (#) in front of the line that starts telnetd . |
| Step 3. | Find the process ID of the inetd process. Enter:

ps -ef grep inetd grep -v grep |
| Step 4. | Send the process the HUP signal. Enter:

kill -1 pid

where <i>pid</i> is the process-ID obtained from Step 3 above. |
| Step 5. | Verify that you cannot telnet to your local host by repeating Step 1. |
| Step 6. | Undo the changes made in Step 2. |
| Step 7. | Repeat Step 3. |
| Step 8. | Repeat Step 1. |

7.3.2 Enabling and disabling daemons

As the previous exercise illustrated, you can enable and disable the starting of daemons by **inetd** by editing the */etc/inet/inetd.conf* file and commenting out or uncommenting the desired entry. Then, **inetd** must be restarted.

You might want to disable some daemons for security reasons. An alternative to disabling the daemons is to set up packet filtering so that only trusted packets make it to these daemons.

For more information on packet filtering and other security-related issues, see Lesson 10: “Network Security.”

7.3.3 Adding new services

You might require networking services to be started by **inetd** that have no entries in the */etc/inet/inetd.conf* file by default. You must make new entries in this file to start these services.

For example, to add a new daemon called *newd*, create an entry that looks like:

```
newd    stream    tcp    nowait    root    /usr/local/bin/newd    newd -x
```

You must also add an entry for **newd** (from the first column) into the */etc/inet/services* file. If **newd** uses TCP port 1023, your entry in */etc/inet/services* looks like:

```
newd    1023/tcp
```

NOTE: Not all Internet service daemons can be started by **inetd**. They have to be specifically written to conform to **inetd**'s protocols. Also, just because you can start a daemon out of */etc/inet/inetd.conf* does not mean that you should. For example, running web servers from *inetd.conf* can have a tremendous impact on your system.

The /etc/inet/inetd.conf File

```
# <service_name> <socket_type> <proto> <flags> <user> <server_pathname> <args>
#
# Ftp and telnet are standard Internet services.
#
ftp      stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.ftpd -a
#ftp     stream  tcp      nowait  root    /usr/sbin/in.ftpd    in.ftpd -a
telnet   stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.telnetd
#telnet  stream  tcp      nowait  root    /usr/sbin/in.telnetd  in.telnetd
#
# Tnamed serves the obsolete IEN-116 name server protocol.
#
name     dgram    udp      wait    root    /usr/sbin/in.tcpd    in.tnamed
#name    dgram    udp      wait    root    /usr/sbin/in.tnamed  in.tnamed
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
shell    stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rshd
#shell   stream  tcp      nowait  root    /usr/sbin/in.rshd    in.rshd
login    stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rlogind
#login   stream  tcp      nowait  root    /usr/sbin/in.rlogind in.rlogind
exec     stream  tcp      nowait  root    /usr/sbin/in.tcpd    in.rexecd
#exec    stream  tcp      nowait  root    /usr/sbin/in.rexecd  in.rexecd
comsat   dgram    udp      wait    root    /usr/sbin/in.tcpd    in.comsat
#comsat  dgram    udp      wait    root    /usr/sbin/in.comsat  in.comsat
talk     dgram    udp      wait    root    /usr/sbin/in.tcpd    in.otalkd
#talk    dgram    udp      wait    root    /usr/sbin/in.otalkd  in.otalkd
ntalk    dgram    udp      wait    root    /usr/sbin/in.tcpd    in.talkd
#ntalk   dgram    udp      wait    root    /usr/sbin/in.talkd   in.talkd
pop-3    stream  tcp      nowait  root    /usr/sbin/in.tcpd    /etc/popper
```

GNAD0706

© The Santa Cruz Operation, Inc.

7.3.4 TCP wrappers and **tcpd(1Mtcp)**

AV 7-6 shows that several of the services in the */etc/inet/inetd.conf* file are started through another intermediate program, **tcpd(1Mtcp)**.

The **tcpd** program provides a facility known as *tcp wrappers*. This facility allows you to control access to service daemons based upon the host that the connection request is coming from.

NOTE: Wrappers are not a firewall. They are a brick in your firewall.

Wrappers are discussed further in Section 10.1, “TCP Wrappers,” on page 10-5.

7.4 Using telnet to test daemons

When you encounter problems with client/server networking software, it is often difficult to tell whether the problem is with the client or the server. If the server can print out diagnostic messages after connecting to or receiving messages from a client, then you can use **telnet** to send the server messages.

To test a specific daemon, you must know which port the daemon is listening on. Recall that port numbers for services are listed in the */etc/inet/services* file. To test a specific daemon running on a specific host:

Step 1. Look up the service's port number in the */etc/inet/services* file.

Step 2. Issue a **telnet** command to the daemon. Enter:

telnet *hostname port*

where *hostname* is the name of the remote host and *port* is the port number obtained in Step 1.

Step 3. Interact with the daemon as appropriate.

Exercise: Testing the local POP3 server

Test the POP3 server on your machine:

Step 1. Log into the system as any user.

Step 2. Find out the port number of the POP3 server. Enter:

grep -i pop3 /etc/inet/services

Step 3. Connect to the server. Enter:

telnet localhost 110

Step 4. Enter this command after the OK message:

user owner

Step 5. Enter this command after the OK message:

pass *passwd*

where *passwd* is the owner's password on your local machine.

Step 6. Enter this command after the OK message:

quit

AV 7-7

Review of r-Commands and TCP Utilities

- | | |
|---------------|--|
| rlogin | Allows users to log into a remote host. If trusted access is set up, no password is needed, otherwise a password is needed. |
| rcp | Allows users to copy files from one host to another. Requires trusted access. |
| rsh | Allows users to execute commands on remote hosts. Requires trusted access. |
| telnet | Similar to rlogin, only a user name and password are almost always required, and trusted access is not consulted |
| ftp | Similar to rcp, only a user name and password is always required and trusted access is not consulted |

GNAD0707

© The Santa Cruz Operation, Inc.

7.5 Configuring Trusted Access

Trusted access allows your users to utilize the enterprise network in a more convenient way.

If trusted access is not configured for the **rlogin**(1tcp) command:

1. It will prompt your users for a password.
2. This password is transmitted across the network, and maybe even the Internet.

Packets containing these passwords are relatively easy to intercept and identify.

If trusted access is set up, no password is necessary for **rlogin**.

If trusted access is not set up, the **rsh**(1tcp) and **rcp**(1tcp) commands do not work.

As a network administrator, your responsibilities include administration of trusted access and *user equivalence*, another network attribute that is explained in the sections below. You can configure trusted access at the host or user level, or both.

AV 7-8

Example */etc/hosts.equiv* File on *corfu*

```
barbados  
tortola bob  
guam
```

GNAD0708

© The Santa Cruz Operation, Inc.

7.5.1 Configuring trusted host access

Trusted host access allows you to indicate that one host trusts all the users on another host that have identical login names to use the following commands without specifying a password:

- **rlogin**
- **rcp**
- **rsh**

You must be **root** to configure trusted access at the host level. This configuration consists of editing the */etc/hosts.equiv* file, creating it if necessary. Each entry in */etc/hosts.equiv* specifies the remote system that is trusted. AV 7-8 illustrates the format of entries in an example */etc/hosts.equiv* file.

If the entries in AV 7-8 exist on a machine called *corfu*, the entries mean the following:

<i>barbados</i>	All non- root users on <i>barbados</i> with the same user name as those on this machine (<i>corfu</i>) are trusted. In other words, jane on <i>barbados</i> can rlogin , rcp , and rsh to <i>corfu</i> as jane without specifying Jane's password on <i>corfu</i> .
<i>tortola bob</i>	Allows bob on <i>tortola</i> to log in as user bob on this machine (<i>corfu</i>).

The **root** account can never be equivalenced by an entry in the */etc/hosts.equiv* file. Refer to *hosts.equiv(4tcp)* for further information.

NOTE: If your network is connected to the Internet, you should always use fully qualified host names in this file for security reasons.

7.5.2 Configuring trusted user access

Trusted user access allows users on one system to indicate which users from another system they trust to use their account and enter these commands without providing a password:

- **rlogin**
- **rsh**
- **rcp**

Unlike trusted host access, trusted user access can be created for users on different machines with different login names.

You can configure trusted user access for your users, or they can do it for themselves. The configuration consists of:

- Step 1. Creating or modifying a file called *.rhosts* (note the leading dot) in a user's home directory.
- Step 2. Setting the ownership so that the file is owned by either **root** or the owner of the home directory.
- Step 3. Setting the permissions so that the file is writable only by the owner.

If steps 2 and 3 are not performed, the file is ignored.

AV 7-9

Example .rhosts File For User **jane** On Host *corfu*

```
barbados  
tortola bob  
guam
```

GNAD0709

© The Santa Cruz Operation, Inc.

The format of entries in the *.rhosts* file is the same as those in the */etc/hosts.equiv* file. The semantics are somewhat different. If the entries in AV 7-9 were put in the home directory of the user **jane** on the machine *corfu*, then they would have the following meaning:

barbados	User jane on <i>barbados</i> is trusted by the user jane on this machine (<i>corfu</i>). In other words, jane on <i>barbados</i> can rlogin , rnp , and rsh to <i>corfu</i> as jane without specifying Jane's password on machine <i>corfu</i> .
tortola bob	User bob on <i>tortola</i> is trusted by the user jane on this machine (<i>corfu</i>). In other words, bob on <i>tortola</i> can rlogin , rnp , and rsh to this machine (<i>corfu</i>) as jane , using the appropriate arguments to those commands, without specifying Jane's password.

Exercise: Setting up trusted access

For this exercise, you will need a partner.

Log in as **guest** on your system. Your instructor will give you the password for the **guest** account.

Remotely log into your partner's system with the following command:
`rlogin partner's_system_name`.

Are you prompted for a password?

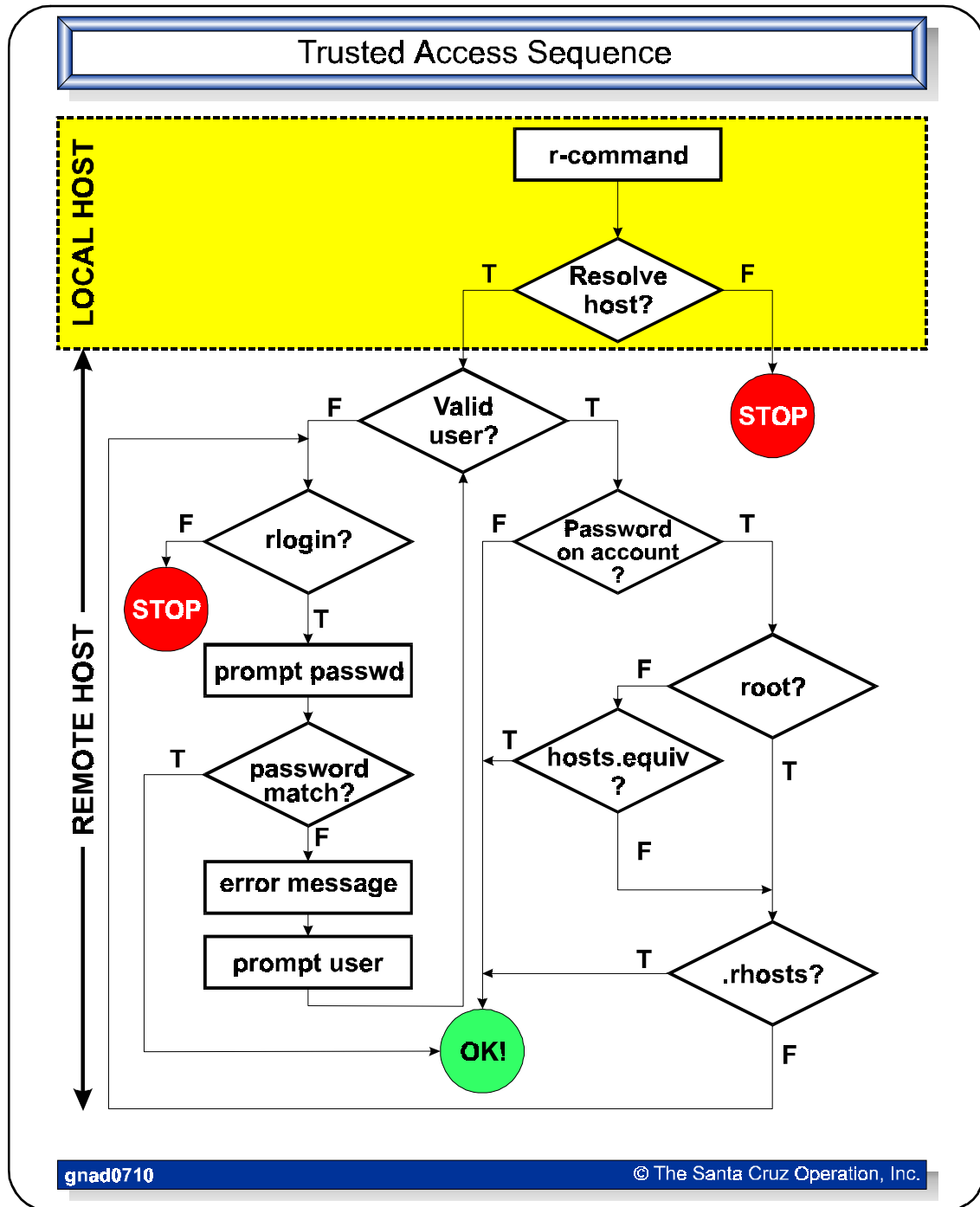
After you have logged in, edit or create a *.rhosts* file in the home directory of **guest** on your partner's machine and add a line with just the name of your system.

Log out and then log in again with the same command from above.

Are you prompted for a password?

Is this an example of host or user trusted access?

AV 7-10



7.5.3 The trusted access sequence

Whenever the **rlogin**, **rsh**, or **rcp** commands are executed, the systems involved go through a set procedure to determine whether trusted access has been configured. AV 7-10 shows this procedure.

7.5.3.1 Trusted access on the local host

These steps describe the trusted access sequence that takes place on the local host:

1. An **r**-command is issued by a user.
2. The local host attempts to resolve the remote host name to obtain its IP address.
3. If the remote host name cannot be resolved into an IP address, the error message, *host_name: Host name lookup failure*, is displayed.
4. If an address for the remote host can be resolved, the **r**-command connects through the TCP/IP transport to the remote host.

7.5.3.2 Trusted access on the remote host

These steps describe the trusted access sequence that takes place on the remote host:

1. The user's account name is looked up in the password database.
2. If the user's account name is not found in the password database, the **rsh** and **rcp** commands fail, and the **rlogin** command prompts for a password (that is how the login program functions), and then fail.
3. The user's account is checked to see if it has a (encrypted) password.
4. If the user's account does not have a password, the **r**-command executes. NOTE: This is very dangerous. You should never allow users to not have passwords on your systems, especially if your network is attached to the Internet.
5. The user's account name is checked to see if the user is **root**.

If the user's account name is not **root**, then */etc/hosts.equiv* is checked for the local host's name.

If the */etc/hosts.equiv* file contains the name of the local host, the **r**-command executes.

6. If the account name check from Step 5 was **root**, or if the */etc/hosts.equiv* check failed, then the *.rhosts* file in the remote user's home directory is checked. If it contains the local host's name, and possibly the local user's name, depending upon how the **r**-command was invoked.
7. If the *.rhosts* file contains the desired entry, then the **r**-command executes.

Failure so far means that the **rsh** and **rcp** commands will fail. If the **r**-command invoked was **rlogin**, it will prompt for the remote user's password, and only succeed if the correct password is given.

AV 7-11

Network Administrator's Account Log

LOGIN	REAL NAME	UID	<i>tortola</i>	<i>guam</i>	<i>barbados</i>	<i>name...</i>
bobb	Bob Byte	990	✓	✓	✓	✓
daven	Dave Network	10057	✓	✓		
ellene	Ellen Ethernet	7431	✓		✓	
karenc	Karen Config	13116	✓		✓	
ricko	Rick Octet	4429	✓	✓		

GNAD0711

© The Santa Cruz Operation, Inc.

7.5.4 Overview of user equivalence

Section 7.5.3, “The trusted access sequence,” on page 7-33 shows that trusted access is based upon user names. What would happen if:

- Two people on different machines have the same user name
- Trusted host access is established between the two hosts

The answer is that the two different users are able to access each others accounts. This is probably not what you want.

The solution to this problem, and related problems, is for network administrators to establish user equivalence for all of the machines in their network.

You establish user equivalence between machines on a network when, for all accounts on the machines in question, you have a one-to-one-to-one correspondence between:

- The real people using a machine
- Their login names (the first field in the */etc/passwd* file)
- Their user IDs (the third field in the */etc/passwd* file)

As the administrator of your system, you should maintain a user account log that tracks account information on all the machines on your network. As shown in the sample form in AV 7-11, such a log should contain:

- User account name and real name
- User ID
- Systems on which a given user has an account

7.5.4.1 Examples of user equivalence

For example, suppose that:

- You have two people, John and Jane, who need to use two machines.
- Each has accounts on two machines *corfu* and *barbados*, with the same account names and user IDs:

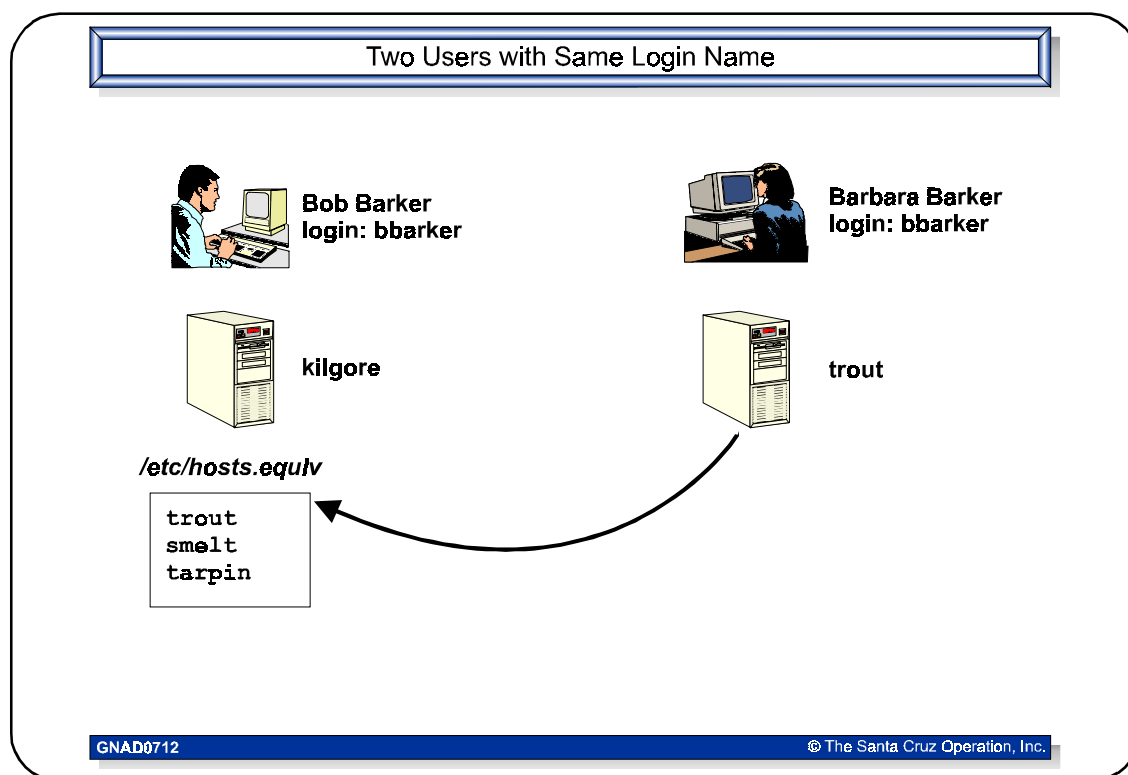
<i>corfu</i>	john , 201 jane , 202
<i>barbados</i>	john , 201 jane , 202

Many network administrators extend the concept of user equivalence to group equivalence. This exists when there is a one-to-one-to-one correspondence between:

- Group names
- Group IDs
- Group members

You need to establish user equivalence if you expect to use trusted access or NFS on your hosts. The maintenance of user equivalence is made much easier with NIS. For more about using NIS, see **scohelp** ⇒ **Networking**.

AV 7-12



7.5.4.2 Trusted access problems solved with user equivalence

The following is an example of problems caused by using trusted access without first having set up user equivalence:

- Bob Barker has an account on his system, which is named *kilgore*. His account name is **bbarker**.
- Barbara Barker has an account on her system, which is named *trout*. Her account name is **bbarker**.
- The system administrator of *kilgore* has configured trusted host access for all other systems in the network in an effort to make the network easier to use for everyone.
- One day Barbara uses the **rlogin** command to login to *kilgore* and finds that she is allowed to login without having to give a password.
- Also, on *kilgore* she has access to all of Bob's files, because on *kilgore* she is Bob (**bbarker**).

For this example, it does not matter if Bob and Barbara have the same UID or not.

The following is an example of problems caused by using NFS without first having set up user equivalence.

- Bob has an UID of 250 on his system which is named *kilgore*.
- Barbara has an UID of 250, as well, on her system which is named *trout*.
- Barbara has all her most personal and sensitive files in a directory which the system administrator of *trout* has made available to network clients (i.e., *exported*) using NFS.

- The system administrator of *kilgore* has mounted the directory containing Barbara's files from *trout*.
- When Bob changes directory to where Barbara's files are mounted and lists the files using the **ls -l** command, it will show him as the owner of the files.

For this example, it does not matter if Bob and Barbara have the same account name or not.

7.5.5 Changing an account's UID and login name

If you find that you have a conflict like those just listed, you need to clearly analyze the problem and carefully create a plan to resolve the conflicts. You can use the **usermod(1M)** command to modify an account's UID and login name. The **usermod(1M)** manual page documents this command fully.

For example, to change the login name of user **bbarker** to **bob**, enter:

```
usermod -l bob bbarker
```

For example, to change the UID of user **bbarker** to 251, enter:

```
usermod -u 251 bbarker
```

NOTE: These commands only change the system's database. You might need to change file and directory ownership as you were taught in the **UnixWare 7 System Administration I: User Services** course.

7.6 Lesson Summary

This lesson discussed the many issues related to TCP/IP administration including startup, shutdown, configuration, the TCP/IP daemons and their function, and trusted access.

Review questions

1. Which files are used to configure trusted access for hosts and users?
2. Will trusted access to an account on a remote machine without a password always fail? Explain.
3. What must be configured for either **r**cp or **r**sh to function?
4. Name four ways that server daemons can be started.
5. Which file lists the server daemons started by **inetd**(1Mtcp)?
6. Which program reads the */etc/inet/config* file?
7. Which signal do you send to **inetd**(1Mtcp) to tell it to re-read its configuration file?

7.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

aasd (1Mtcp)	The address allocation server daemon
bootpd (1Mtcp)	The boot protocol server daemon
dhcpcd (1Mtcp)	The dynamic host configuration protocol server daemon
ftp (1tcp)	TCP/IP standard file copy program
gated (1Mtcp)	A multi-protocol routing daemon
inetd (1Mtcp)	The super daemon (it starts many other daemons on demand)
ifconfig (1Mtcp)	Network interface configuration
named (1Mtcp)	The name server daemon
ndcfg (1M)	Network driver configuration
ndstat (1M)	Displays network adapter statistics
rlogin (1tcp)	Remote login to another UNIX machine
rlogind (1Mtcp)	Server program for rlogin (1tcp)
rsh (1tcp)	UNIX specific remote command
rcp (1tcp)	UNIX specific remote copy
routed (1Mtcp)	A RIP-only routing daemon
scohelp (1X)	Online help and documentation
slink (1Mtcp)	Configures STREAMS and socket
snmpd (1Mtcp)	Simple network management protocol daemon
syslogd (1Mtcp)	Logs system messages to <i>/usr/adm/syslog</i> by default

telnet (1tcp)	TCP/IP standard remote login
tcpd (1Mtcp)	Implements TCP wrappers
usermod (1M)	Modifies user accounts
xntpd (1Mtcp)	Standard time daemon

Relevant files

<i>~/.rhosts</i>	Is the configuration file for user trusted access
<i>/etc/confnet.d/inet/interface</i>	Stores the network interfaces and their associated parameters
<i>/etc/hosts.equiv</i>	Is the configuration file for host trusted access
<i>/etc/inet/config</i>	Is the configuration file for the <i>/etc/confnet.d/inet/initialize</i> command. Contains a list of programs to run as part of TCP/IP startup. Documented in <i>config</i> (4tcp).
<i>/etc/inet/gated.conf</i>	Is the configuration file for gated (1Mtcp)
<i>/etc/inet/gateways</i>	Is the configuration file for routed (1Mtcp)
<i>/etc/inet/inetd.conf</i>	Is the configuration file for the inetd (1Mtcp) daemon. Contains a list of daemons inetd should start on demand. Documented in <i>inetd.conf</i> (4tcp).
<i>/etc/inet/macaddr</i>	Stores the hardware address of each network adapter on the system.
<i>/etc/inet/strcf</i>	Is the STREAMS configuration file used by slink (1Mtcp)
<i>/etc/inet/services</i>	Is a list of names for server daemons, their protocol, and their port number. Documented in <i>services</i> (4tcp).
<i>/usr/sbin/initialize</i>	Contains the program that helps initialize the TCP/IP protocol stack and reads the <i>/etc/inet/config</i> file.
<i>/var/adm/syslog</i>	Is the default location for output from syslogd (1Mtcp)

7.8 Answers to Review Questions

1. The */etc/hosts.equiv* and *.rhosts* files in the user's home directory are used to configure host and user trusted access, respectively.
2. Trusted access to an account on a remote machine without a password does not always fail. The *rlogin* command always succeeds to accounts without passwords.
3. Trusted access must be configured for either **r**cp or **r**sh to function. The shell entry in */etc/inet/inetd.conf* must also be configured.
4. Server daemons can be started in any of the following ways:
 - Through an entry in a system startup script
 - Through an entry in the */etc/inet/config* file
 - Through an entry in the */etc/inet/inetd.conf* file
 - Through the Service Access Facility (SAF)
5. The */etc/inet/inetd.conf* file lists the server daemons started by **inetd**(1Mtcp).
6. The */etc/inet/config* file is read by the */usr/sbin/initialize* program during TCP/IP startup.
7. You would send **inetd**(1Mtcp) signal 1, the HUP signal, to tell it to re-read its configuration file.

8

Domain Name System (DNS) Configuration

AV 8-1

Domain Name System (DNS) Configuration

- DNS operation
- Configuring name servers
- Querying a name server
- Configuring DNS clients

GNAD0801 © The Santa Cruz Operation, Inc.

Lesson 8

Domain Name System (DNS) Configuration

Introduction

Lesson 2: “TCP/IP Concepts” discussed the resolution of IP addresses to host names using the `/etc/hosts` file and other alternatives. These alternatives include the use of the Network Information Service (NIS), discussed in **scohelp** ⇒ **Networking** ⇒ **Administering the Network Information Service**; and the Domain Name Server (DNS) which is covered in this lesson.

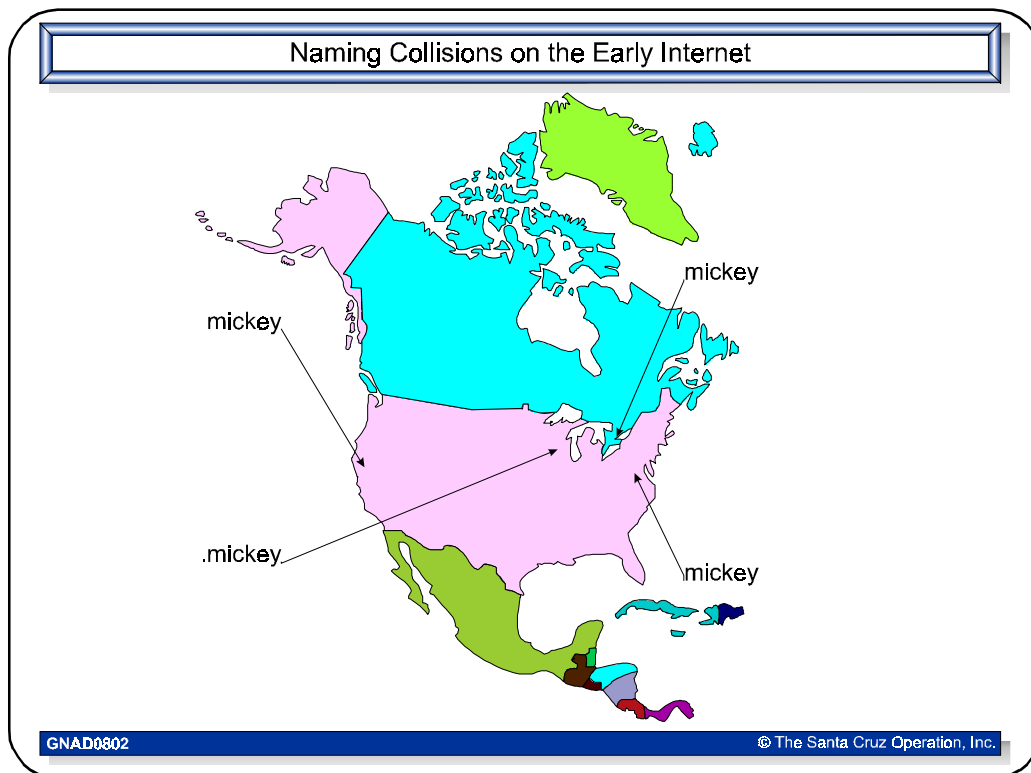
You must configure DNS when connecting your enterprise network to the Internet. Configuring DNS is also part of a natural progression as your enterprise network grows beyond a few machines. When you tire of updating the `/etc/hosts` file on every machine in your network, it is time to set up DNS.

Objectives

This lesson describes how to set up DNS on machines running UnixWare 7. By the end of this lesson, you will be able to:

- Describe what DNS is and how it works
- Create a database of DNS information for a domain using the various DNS record types in standard resource record (SRR) format
- Configure a primary, secondary, and caching-only name server
- Configure a DNS client
- Use **nslookup**(1Mtcp) to query a name server’s database

AV 8-2



8.1 DNS Operation

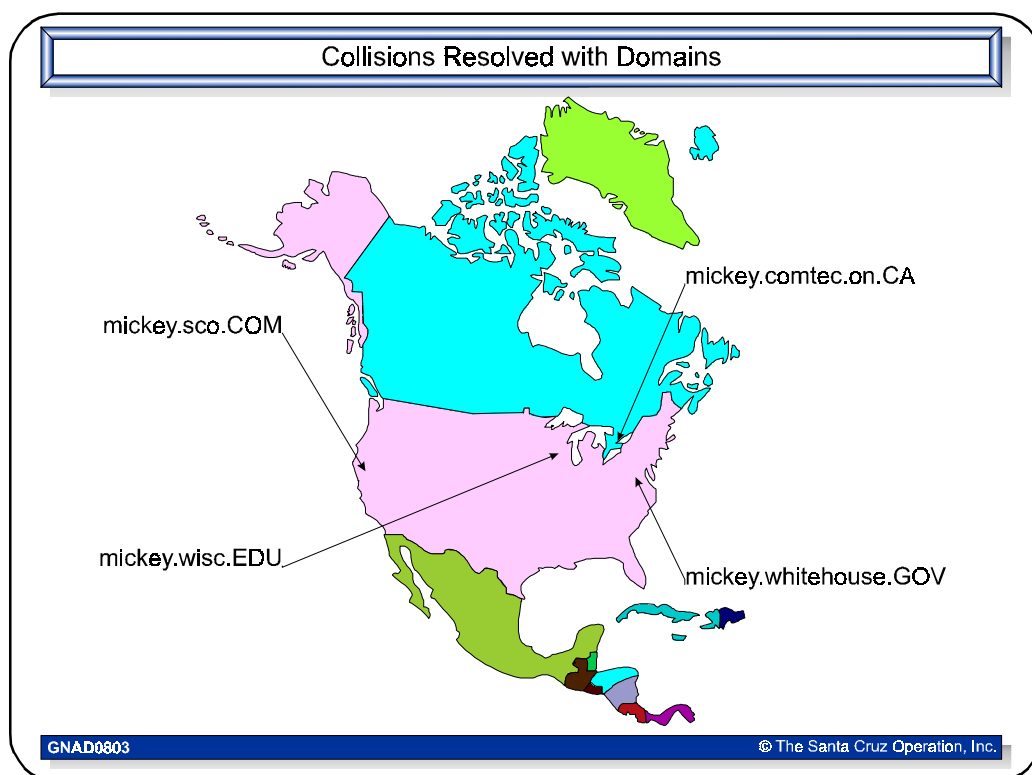
When the Internet was initially established:

- It consisted of fewer than 100 machines.
- The agency controlling the Internet kept a single file that had the name and IP address of every machine on the Internet. Network administrators downloaded this file to their machine nightly. On UNIX[®] based machines, this file became the */etc/hosts* file.

When the Internet grew to hundreds of machines, several problems with this approach appeared:

- If new machines were added daily, there were occasional collisions. That is, two sites on the Internet chose the same machine name. Someone had to decide who got to keep their choice and who had to choose again.
- When many machines were being added daily, the people maintaining the */etc/hosts* file could not keep up.
- The file started to grow so large that a linear search made by the application software through the */etc/hosts* file was taking a significant amount of time.

AV 8-3



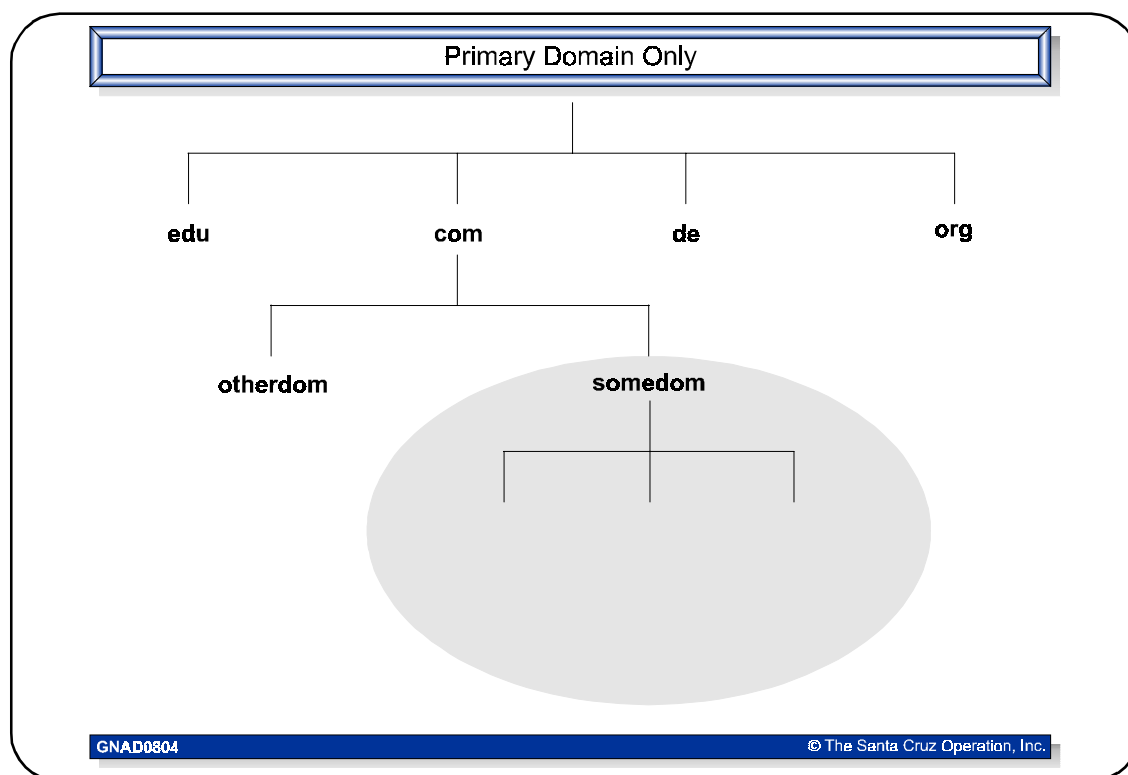
The Internet community proposed many solutions to the problems above. Some solutions solved only part of the problem, some were very difficult to deal with, and some did not work.

The solution eventually decided upon was the *Domain Name Service* (DNS). The DNS:

- Was proposed by the University of California at Berkeley
- Is called Berkeley Internet Name Domain (BIND) service
- Has a daemon that implements BIND on SCO systems running the name server called **named**(1Mtcp), the name daemon.

The **named** daemon listens for requests from clients anywhere on the Internet to resolve names into IP addresses and vice-versa. Additional functionality is covered later in this lesson.

AV 8-4



8.1.1 DNS server types

DNS supports these server configuration types:

- *Primary* name servers, also referred to as *master* name servers
- *Secondary* name servers, also referred to as *slave* name servers
- *Caching-only* name servers

The master/slave terms are introduced in UnixWare 7.1 with the implementation of BIND version 8. However, because the terms primary and secondary are in common usage throughout the computer industry, they are used interchangeably with master and slave in this lesson.

8.1.1.1 Primary name servers (PNS)

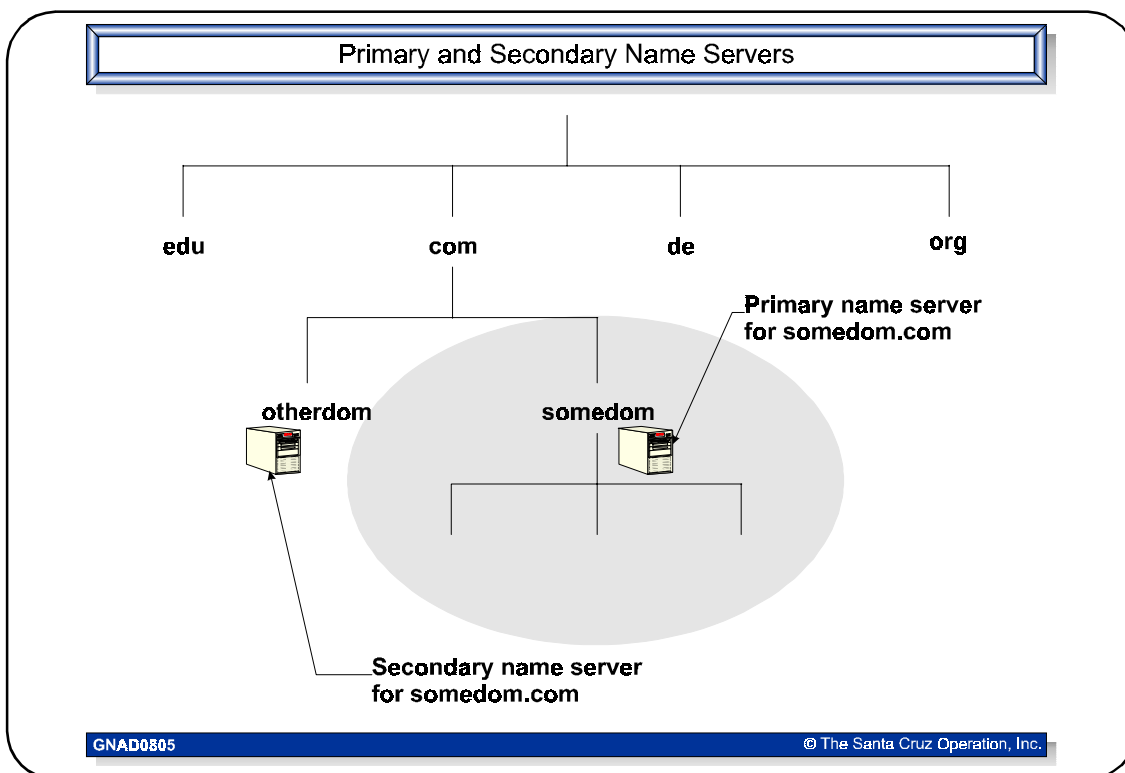
The information in its configuration file can tell **named** that it is a primary name server (PNS) for some portion of the domain tree. Being a PNS means that **named** on this host is *the* authority for this portion of the domain tree. The rest of the Internet should contact this host when trying to resolve domain names in this portion of the domain tree.

The set of machines that PNS has authority over is typically a whole domain, however it could be a subdomain. The area of the domain tree that a name server has authority over is called its *zone of authority*.

Every domain on the Internet must have one registered primary name server and there should be only one primary for each domain.

In AV 8-4, a single name server presides over the entire *somedom.com* domain. Whenever anyone on the Internet wants to resolve a host name or IP address in this domain, they must contact this name server.

AV 8-5



8.1.1.2 Secondary Name Servers (SNS)

If you only set up a PNS for your network, you can experience two problems:

- If the primary name server goes down for some reason, none of the Internet clients can resolve machine names or IP addresses. This often results in the client failing to function.
- It might become a bottleneck for the Internet clients trying to resolve machine names or IP addresses.

A solution to both of these issues is to set up a *secondary name server* (SNS).

A zone:

- Should have only one *primary* name server
- Can have multiple *secondary* name servers. These secondary name servers can be queried by Internet clients either when the primary is down or to relieve the primary server of some of its load.

Primary name servers assign a version number to the data currently serving. This number is called a *serial number* and must be increased every time you change the data.

Each domain on the Internet must have at least two registered name servers: one primary and one secondary.

The following steps illustrate how primary and secondary name servers interact. The **named** daemon:

1. Starts running on a host in *otherdom.com* and determines from its configuration file that it is a SNS for the *somedom.com* domain.
2. Reloads the data if it was directed by the configuration file to store a copy of the zone's data on disk. The data also indicates when it was last validated and what its serial number is.
3. Attempts to contact the PNS to see if the serial (version) number is still the same. If the PNS:

Does not answer, **named** on the SNS continues to try.

Continues to not answer, eventually the data is marked as invalid (**named** on the SNS will not serve it to clients) and **named** on the SNS may even give up trying to contact the PNS.

4. Asks the PNS for a copy of all of the data for that zone if the serial number is different. This is called a *zone transfer*.

If the serial number is the same, then no zone transfer is needed and the data is stamped with the current time.

The parameters controlling this behavior are:

<i>retry</i>	How often to retry
<i>minimum</i>	How long to keep the data
<i>expire</i>	When to give up retrying

All of these parameters are stored in the primary name server's database and kept by the secondary whenever a successful contact is made. See Section 8.2.7.5, "The SOA record and `$include`," on page 8-47 for further details on these parameters.

8.1.1.3 Caching-only servers

All name servers:

- Can be configured to forward queries for information outside their zone to other name servers.
- Cache answers to queries for the time specified by the *minimum* parameter set on the PNS.

This cuts down on the need to forward common requests and also speeds up service to clients.

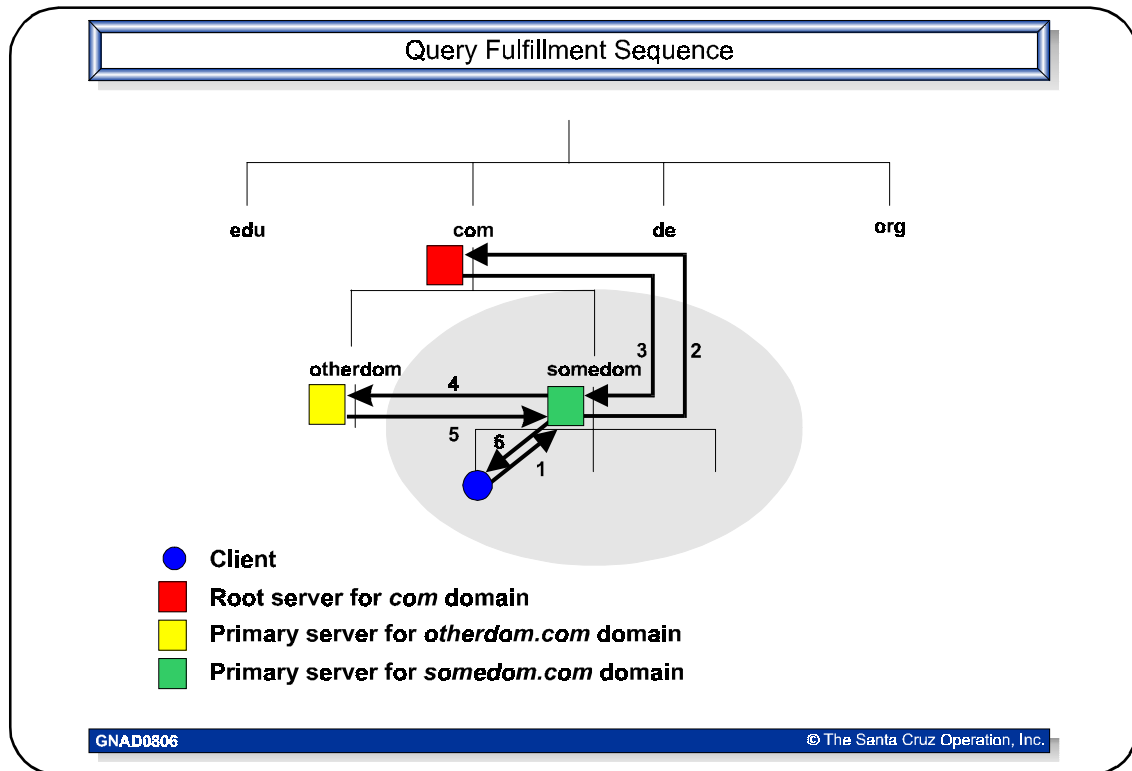
Caching-only name servers are different from other name servers in that they:

- Do not maintain data files for a zone
- Do not do zone transfers from a PNS
- Only get data on demand from a PNS or SNS

Unless your system has limited memory or CPU power, you might achieve the best performance by setting your system up as a caching only name server instead of just as a client. This reduces network traffic and DNS response time for frequently requested names.

You can configure a machine as a caching-only name server only if it is not running any other type of name server. Alternatively, if a caching-only name server has previously been configured, it must be deleted before other name servers can be configured.

AV 8-6



1. An Internet client (*client.somedom.com*) makes a library call to look up the IP address for a machine name (*remote.otherdom.com*).

The library routine detects the */etc/resolv.conf* file indicating the machine's current domain and the IP address of the local name server (*dns.somedom.com*).

The library routine sends the request to *dns.somdom.com*.

2. The local name server notes that the request is for information outside its zone.

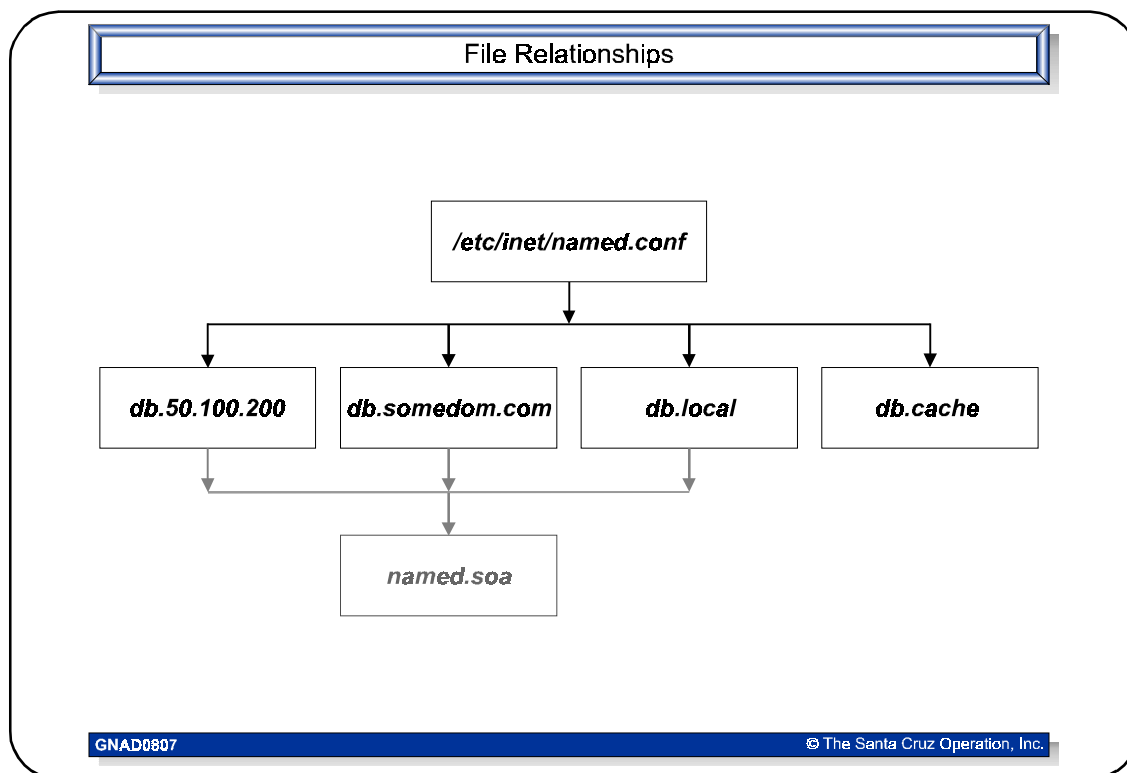
It looks up the IP address for a root name server and contacts the root name server to request the IP address of the name server for the *otherdom.com* domain.

3. The root name server sends back the IP address of *dns.otherdom.com*, the name server for *otherdom.com*.
4. *dns.somdom.com* queries *dns.otherdom.com* for the IP address of the target, *remote.otherdom.com*.
5. *dns.otherdom.com* sends a reply with the IP address of *remote.otherdom.com*.
6. The reply is sent back to the client, *client.somedom.com*.

The top-level server contacted is called a *root name server*. A root name server is authoritative for all the top-level domains, such as *com*, *gov*, and *edu*. This means that it is essential for resolving an Internet domain name.

Because a root name server plays such a crucial role in DNS operations, there are actually several root name servers spread across the globe. A name server knows the IP addresses of the root name servers through the use of a *root cache* file. The file is discussed later in this lesson. To keep a name server running optimally, it is prudent to update this file frequently.

AV 8-7



8.2 Configuring DNS Name Servers

This lesson has covered what a name server is and how it works, including what primary, secondary, and caching-only name servers are and how queries for information outside a zone get fulfilled. Now, you are ready to configure name servers.

AV 8-7 shows the relationships between the files. The names of these files are configurable and can be different on specific systems.

The names listed are examples of names the **DNS Manager** generates for these files:

<i>named.conf</i>	The file used to configure the named (1Mtcp) daemon on start up, unless the -b flag is specified. This file points to the other data files.
<i>db.somedom.com</i>	A file providing translations for host names in the <i>somedom.com</i> domain to IP addresses.
<i>db.50.100.200</i>	A file providing translations of IP addresses in the 200.100.50 network into host names.
<i>db.local</i>	A file providing a translation of IP addresses in the 127.0.0 network into host names. The only entry is for localhost.
<i>db.cache</i>	A file containing the cache of root-level name servers.

The *named.soa* file is often used when setting the files up by hand to provide a single point of configuration for SOA resource values, in particular the serial number. In AV 8-7, this box is grey because the **DNS Manager** does not use an SOA file but instead writes SOA records at the top of each file.

Note for those familiar with previous releases of BIND which use the *named.boot* configuration file, the conversion utility **named-bootconf.pl** utility can be used to create a *named.conf* file from an existing *named.boot* file. As **root**, enter the following command to perform the conversion:

```
/usr/sbin/named-bootconf.pl /etc/inet/named.boot > /etc/inet/named.conf
```


8.2.1 How **named** works

The **named** daemon is typically started by the `/usr/sbin/initialize` program as directed by the following entry in the `/etc/inet/config` file:

```
5:/usr/sbin/in.named::y:/etc/inet/named.conf::
```

Startup of Internet services is discussed in Section 7.2.1, “The `config(4tcp)` file,” on page 7-13.

The exact syntax for starting **named** can be found in the **named(1Mtcp)** manual page.

During **named** startup:

1. **named** looks for its control file, called `/etc/inet/named.conf` (this can be changed with a command-line parameter). If this file is:

Found, **named** reads and acts on the contents of the file.

Not found, **named** exits.

2. The contents of the `/etc/inet/named.conf` file point **named** to a set of data it loads into its internal tables. This data is usually located in files found in the `/etc/inet/named.d` directory.
3. **named** listens for queries for host names and addresses from various clients and answers them from the data it loaded in the previous step.

Example *named.conf*

```
options {
    directory "/etc/inet/named.d";
};

zone "somedom.com" in {
    type master;
    file "db.somedom.com";
};

zone "50.100.200.in-addr.arpa" in {
    type master;
    file "db.50.100.200";
};

zone "0.0.127.in-addr.arpa" in {
    type master;
    file "db.local";
};

zone "otherdom.com" in {
    type slave;
    file "db.other.com";
    masters {
        200.150.75.1;
    };
};

zone "75.150.200.in-addr.arpa" in {
    type slave;
    file "db.75.150.200";
    masters {
        200.150.75.1;
    };
};

zone "london.somedom.com" in {
    type stub;
    file "db.london.stub";
    masters {
        200.200.50.1;
    };
};

zone "." in {
    type hint;
    file "db.cache";
};
```

GNAD0808

© The Santa Cruz Operation, Inc.

8.2.2 The *named.conf* file

As previously stated, when **named**(1Mtcp) is started, it looks for a file called */etc/inet/named.conf*. AV 8-8 and the following text describe entries that can be found in the *named.conf* file for a server with the following attributes:

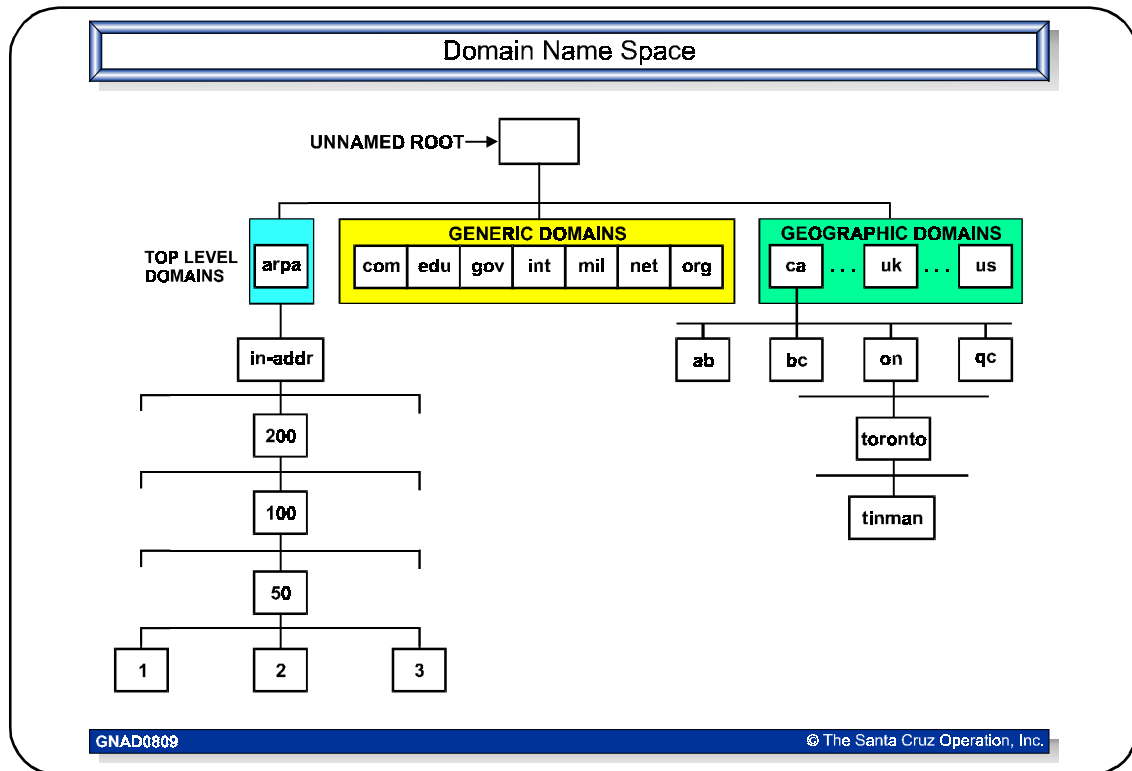
- It acts as a primary name server for the *somedom.com* domain.
- It is a secondary server for the *otherdom.com* domain.
- The *somedom.com* domain has a network IP address of 200.100.50.
- The *otherdom.com* domain has a network IP address of 200.150.75.
- The *london.somedom.com* domain is served by a stub server at the IP address of 200.200.50.1.

8.2.2.1 The directory option

The **directory** option tells **named** the directory in which the files referenced by this file are located. It is usually set to either:

- */etc/inet/named.d* (UnixWare 7)
- */etc/named.d* (SCO OpenServer Release 5)
- */usr/lib/named* (previous releases of SCO UNIX)

AV 8-9



8.2.2.2 The master type

The master type tells **named**:

- It is the primary name server for the indicated zone.
- The data files for the hosts in that zone can be found in the filename(s) in the **file** field.

At least three master entries are needed for a primary name server. The separate entries are for:

- The zone data by name, that is, all hosts under *somedom.com*.
- Your entry in the *in-addr.arpa* domain, which allows the lookup of host names by IP address. Without this hierarchy, a DNS server would need to conduct an exhaustive search of the domain tree looking for a given IP address. Because the *in-addr.arpa* domain is organized by IP address, not domain name, it makes possible efficient lookups of host names from IP addresses.

The IP address in this entry is written backwards. To see why this format is necessary, consider:

- Fully-qualified host names are written from left to right with the least significant piece of information, (the lowest part of the domain tree), first. This allows for quick lookup of IP addresses by machine name, as previously discussed.
- IP addresses are written from left to right with the most significant piece of information first—the network number. If you wanted to look up a host name given its IP address, the information would be in the wrong order.

To solve this problem, in the *in-addr.arpa* domain, IP addresses are written in reverse order. AV 8-9 illustrates entries in the *in-addr.arpa* domain for the first few addresses in the *somedom.com* domain.

- The IP address of the *localhost* interface in reverse order.

AV 8-10

Example *named.conf*

```
options {
    directory "/etc/inet/named.d";
};

zone "somedom.com" in {
    type master;
    file "db.somedom.com";
};

zone "50.100.200.in-addr.arpa" in {
    type master;
    file "db.50.100.200";
};

zone "0.0.127.in-addr.arpa" in {
    type master;
    file "db.local";
};

zone "otherdom.com" in {
    type slave;
    file "db.other.com";
    masters {
        200.150.75.1;
    };
};

zone "75.150.200.in-addr.arpa" in {
    type slave;
    file "db.75.150.200";
    masters {
        200.150.75.1;
    };
};

zone "london.somedom.com" in {
    type stub;
    file "db.london.stub";
    masters {
        200.200.50.1;
    };
};

zone "." in {
    type hint;
    file "db.cache";
};
```

GNAD0810

© The Santa Cruz Operation, Inc.

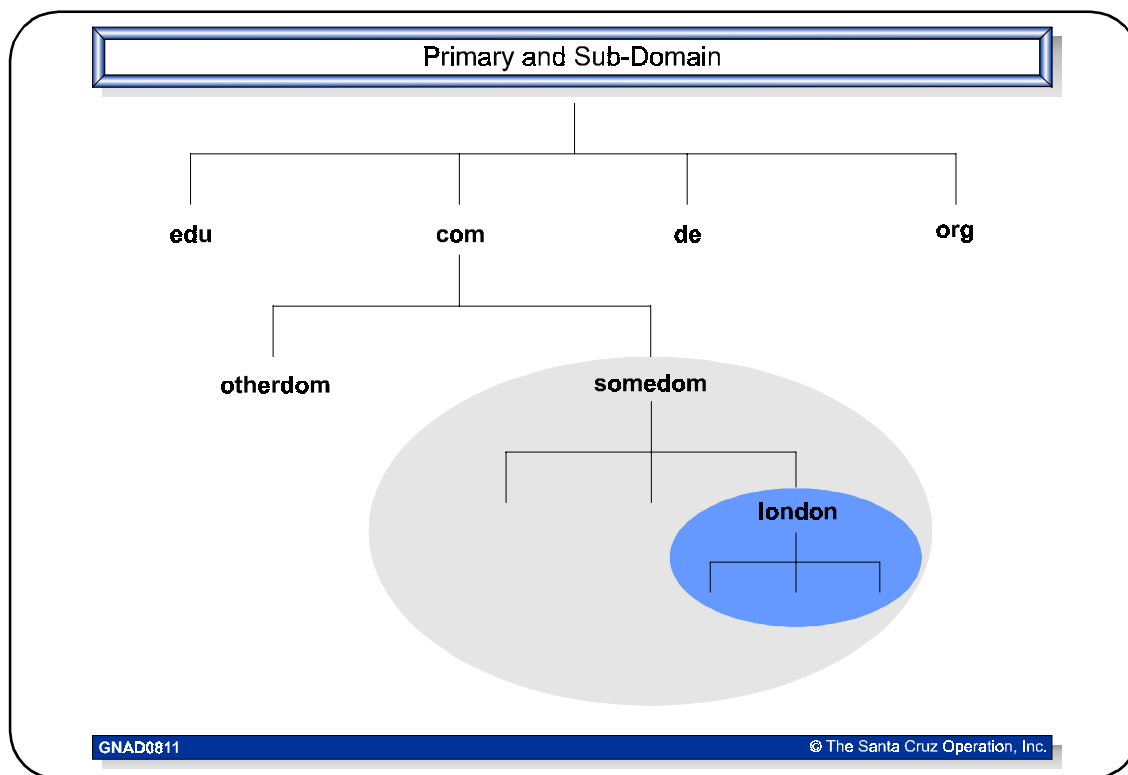
8.2.2.3 The `slave` type

The `slave` type tells **named**:

- It is the secondary name server for the indicated zone.
- The IP address of the primary server for the zone is in the `masters` field.
- Optionally, the data for that zone is stored in the file specified in the `file` field. If a filename is:
 - Specified, then the name server loads the data from disk first, and then contacts the primary(s) to make sure the data is up to date using the *serial number* from the primary's start of authority record.
 - Not specified, the PNS must perform a zone transfer before **named** on the SNS can answer queries. Remember that the data file is only a backup. The primary server always has the original, real data.

NOTE: As with PNSs, each secondary server must have two `slave` name server entries per zone: one to resolve lookups by name, and one to resolve lookups by IP address.

AV 8-11



8.2.2.4 The `stub` entry

The `stub` entry tells **named**:

- A separate server is the PNS for a portion of a zone served by this system.
- The name of this sub-domain is specified in the `zone` field.
- The IP address of the primary server for the sub-domain is in the `masters` field.
- Optionally, the data for the sub-domain is stored in the file specified in the `file` field. If a filename is:
 - Specified, then the name server loads the data from disk first, and then contacts the primary to make sure the data is up to date, using the *serial number* from the primary's start of authority record.
 - Not specified, the PNS must perform a zone transfer before **named** can refer queries for this sub-domain. Remember that the data file is only a backup. The primary server always has the original, real data.

In AV 8-11, a primary name server is covering the *somedom.com* domain, except for the *london.somedom.com* domain which is handled by a different name server.

To do this, the primary name server for *somedom.com* must delegate authority over the child zone *london.somedom.com* by including a `stub` entry for the child zone.

AV 8-12

Example Cache File

```
; name          ttl      class  type  data
; initial cache data for root domain servers.  Note no SOA included

.                999999  IN     NS   terp.umd.edu.
.                999999  IN     NS   ns.nasa.gov.
.                999999  IN     NS   kava.nisc.sri.com.

; Prep (hardwire) the address cache.  Order does not matter.

terp.umd.edu.    999999  IN     A    128.8.10.90
ns.nasa.gov.     999999  IN     A    128.102.16.10
ns.nasa.gov.     999999  IN     A    128.52.195.10
kava.nisc.sri.com. 999999  IN     A    192.33.33.24
```

GNAD0812

© The Santa Cruz Operation, Inc.

8.2.2.5 The cache entry

The cache entry tells **named**:

- To initialize its cache with data for the indicated zone
- That the file that contains the cache data is in the `file` field

The domain specified in the second field is almost always the *root* domain (`.`), with data found in the file specified in the last field.

The entries you place in the data file are typically NS and A records for the top level name servers.

The *db.cache* file, shown in AV 8-12:

- Is referenced from the entry in *named.conf*
- Contains A records for all of the root name servers
- Contains NS records indicating that each of these machines are name servers for the root domain (`.`)

Updates can be performed by selecting **Options** ⇒ **Update Cache File** from the **DNS Manager**. The **DNS Manager** is discussed in Section 8.2.8, “The DNS Manager,” on page 8-51.

8.2.2.6 The forwarders option

The `forwarders` option is not present in the example. When present, it:

- Tells **named** to forward requests to one or more other name servers when it does not know the answer
- Lists the IP addresses for each server to forward requests to, separated by semi-colons
- Is often used when security measures, such as firewalls or packet filters, prevent clients from reaching the Internet name servers directly
- Can be used to increase the performance of local name service resolution by keeping a centralized nameserver cache

Forwarders send recursive queries as clients do instead of performing iterative queries like standard name servers. After a short period of time, forwarders resort to normal server operation if resolution has not succeeded.

Multiple forwarding addresses can be specified. These addresses:

- Are queried in the order specified.
- Can be the same address appearing multiple times. This increases the time the forwarder allows the server before giving up.

8.2.2.7 The forward-only option

The `forward-only` option:

- Is typically only found in a secondary or caching-only server.
- Tells **named** that it can *only* use a forwarding server to answer queries. This stops the forwarder from resorting to normal operation if the contacted name servers fail.

This is useful when the server being configured has no direct connection to the Internet (the connection is through a firewall or proxy server).

Because it will not resort to normal operation, it might be useful to specify multiple instances of the forwarding addresses to increase the time-out before failure.

AV 8-13

Example *db.somedom.com*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com. steve.somedom.com. (
                                9603110830; serial
                                10800;      refresh
                                1800;      retry
                                3600000;   expire
                                86400 );    minimum (default ttl)

thatch      IN      A        200.100.50.1
www         IN      CNAME    thatch

guam        IN      A        200.100.50.2
ftp         IN      CNAME    guam

atlantis    IN      A        200.100.50.3

gateway     IN      A        200.100.50.4
```

GNAD0813

© The Santa Cruz Operation, Inc.

AV 8-14

Example *db.50.100.200*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com. steve.somedom.com. (
                                9603110830; serial
                                10800;      refresh
                                1800;      retry
                                3600000;   expire
                                86400 );    minimum (default ttl)

1           in      ptr      thatch.somedom.com.
2           in      ptr      guam.somedom.com.
3           in      ptr      atlantis.somedom.com.
4           in      ptr      gateway.somedom.com.
```

GNAD0814

© The Santa Cruz Operation, Inc.

8.2.3 The *db.somedom.com* file

The *db.somedom.com* file, shown in AV 8-13, maps host names to IP addresses. It typically includes:

- Start of authority information (or alternately, a reference to a *named.soa* file)
- One A record for each host in the zone
- CNAME records for creating aliases or nicknames for hosts

See Section 8.2.7.1, “The A record,” on page 8-39 and Section 8.2.7.2, “The CNAME record,” on page 8-41 for details on these records.

8.2.4 The *db.50.100.200* file

The *db.50.100.200* file, shown in AV 8-14, maps IP addresses to host names. It includes:

- Start of authority information (or alternately, a reference to a *named.soa* file)
- One PTR record for each host in the zone

See Section 8.2.7.3, “The PTR Record,” on page 8-43 for details on the format of PTR records.

AV 8-15

Example *db.local*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com. steve.somedom.com. (
                                9603110830; serial
                                10800;    refresh
                                1800;     retry
                                3600000;  expire
                                86400 );   minimum (default ttl)

1           IN      PTR      localhost.
```

GNAD0820

© The Santa Cruz Operation, Inc.

AV 8-16

Example Cache File

```
; name          ttl      class type data
; initial cache data for root domain servers.  Note no SOA included

.                999999  IN      NS      terp.umd.edu.
.                999999  IN      NS      ns.nasa.gov.
.                999999  IN      NS      kava.nisc.sri.com.

; Prep (hardwire) the address cache.  Order does not matter.

terp.umd.edu.    999999  IN      A       128.8.10.90
ns.nasa.gov.     999999  IN      A       128.102.16.10
ns.nasa.gov.     999999  IN      A       128.52.195.10
kava.nisc.sri.com. 999999  IN      A       192.33.33.24
```

GNAD0816

© The Santa Cruz Operation, Inc.

8.2.5 The *db.local* file

The *db.local* file, shown in AV 8-15, translates IP address 127.0.0.1 to *localhost*. It includes:

- Start of authority information (or alternately, a reference to a *named.soa* file)
- One PTR record for *localhost*

See Section 8.2.7.3, “The PTR Record,” on page 8-43 for details on the format of PTR records.

8.2.6 The *db.cache* file

The *db.cache* file, shown in AV 8-16, contains information about the name servers for the root domain. It contains:

- A records for all of the root name servers
- NS records indicating that each of these machines are name servers for the root domain (.)

These records are described in more detail in Section 8.2.7.1, “The A record,” on page 8-39 and Section 8.2.7.4, “The NS record,” on page 8-45.

The *db.cache* file should be updated periodically to the current version. It is available using anonymous **ftp** from *ftp.rs.internic.net*.

AV 8-17

Example DNS Records

```
; name [ttl] class type data
thatch      in    a      200.100.50.1
ftp         in    cname  guam.somedom.com.
1           in    ptr    thatch.somedom.com.
somedom.com. in    ns     thatch.somedom.com.
@           in    soa     thatch.somedom.com. steve.somedom.com. (
                        9603110830; serial
                        10800;  refresh
                        1800;   retry
                        3600000; expire
                        86400 ); minimum (default ttl)
```

GNAD0817

© The Santa Cruz Operation, Inc.

8.2.7 DNS Records

Primary name servers read data for their zone from disk files. The records in these files are in a format known as the *standard resource record* (SRR) format.

All of the records that you create for your name server must comply with this format or the name server might not work at all. Refer to *named.hosts(4tcp)* for more information on record formats.

AV 8-17 shows examples of various records in the SRR format. The next sections describe these records using the *somedom.com* example.

In */etc/inet/named.conf*, the **primary** entry for the *somedom.com* zone indicates that the data for that zone can be found in the *db.somedom.com*, *db.50.100.200*, *db.local* and *db.cache* files.

AV 8-18

Example *db.somedom.com*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com.  steve.somedom.com.  (
                                9603110830; serial
                                10800;      refresh
                                1800;      retry
                                3600000;   expire
                                86400 );    minimum (default ttl)

thatch      IN      A        200.100.50.1
www         IN      CNAME    thatch

guam        IN      A        200.100.50.2
ftp         IN      CNAME    guam

atlantis    IN      A        200.100.50.3

gateway     IN      A        200.100.50.4
```

GNAD0818

© The Santa Cruz Operation, Inc.

8.2.7.1 The A record

A (address) records give the IP address for a host. Set the fields of these records as follows:

name	Set to the host name
time-to-live	Typically left blank
address class	Set to IN for Internet
record type	Set to A
record-specific data	The IP address of the host

NOTE: You should have one **A** record for every machine in a zone, plus one for the *localhost*.

8.2.7.2 The CNAME record

CNAME (canonical name) records:

- Allow you to create aliases or nicknames for your hosts
- Can be very useful if:
 - Your site provides services such as a web server or a FTP server.
 - You do not want to dedicate separate machines to performing those tasks just so that they can be called *www.yourdom.com* and *ftp.yourdom.com*.

You can accomplish the same thing by entering multiple A records for a given IP address but then no name would be the *real* name for this host and some services such as **sendmail** may have a problem with these multiple names.

CNAME records should have their fields set as follows:

“name”	Contains the nickname
“time to live”	Typically left blank
“address class”	Set to IN for Internet
“record type”	Set to CNAME
“record-specific data”	Contains the real host name

As well as having multiple names for one IP address, it might be useful to configure multiple IP address for one name. This can be:

- Done by entering multiple A records for each IP address
- Useful when you have a number of servers fulfilling the same function

The name server when queried for this name returns all the applicable addresses in the order determined by the name server.

AV 8-19

Example *db.50.100.200*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com.  steve.somedom.com. (
                                9603110830; serial
                                10800;      refresh
                                1800;      retry
                                3600000;   expire
                                86400 );    minimum (default ttl)

1           IN      PTR      thatch.somedom.com.
2           IN      PTR      guam.somedom.com.
3           IN      PTR      atlantis.somedom.com.
4           IN      PTR      gateway.somedom.com.
```

GNAD0819

© The Santa Cruz Operation, Inc.

AV 8-20

Example *db.local*

```
; name [ttl] class type data
@           in      soa      thatch.somedom.com.  steve.somedom.com. (
                                9603110830; serial
                                10800;      refresh
                                1800;      retry
                                3600000;   expire
                                86400 );    minimum (default ttl)

1           IN      PTR      localhost.
```

GNAD0820

© The Santa Cruz Operation, Inc.

8.2.7.3 The PTR Record

PTR (for pointer) records are:

- Found in the *in-addr.arpa* domain files such as *db.50.100.200*
- Typically the only records found there besides an SOA. These records map an IP address to a host name.

You should have one PTR record per host that shares the IP network number listed in the second field of the entry in the *named.conf* file that referenced this file.

PTR records should have their fields set as follows:

“name”	Should contain the host portion of the IP address, in reverse order (for standard class C addresses, only put the last octet; for standard class B addresses, put the last two octets)
“time to live”	Typically left blank
“address class”	Set to IN for Internet
“record type”	Set to PTR
“record-specific data”	Contains the fully qualified host name ending in a period

The *db.local* file includes an SOA and a single PTR record for the local host. AV 8-20 illustrates what a *db.local* file should look like.

8.2.7.4 The NS record

NS records provide information about which hosts are providing names services for specified domains. Set the fields of an NS record as follows:

“name”	Should contain the name of the domain
	The domain name can be a period (.) indicating that these are name servers for the root domain. For example, the period (.) is used in the <i>db.cache</i> file.
	The domain name can be an (@) indicating that these are name servers for the current domain. For example, this is often used in the SOA record.
	The domain name could be blank, indicating the this record should use the same domain name as the previous record.
“time to live”	Left blank in the <i>named.soa</i> file entries. Set to 999999 in <i>db.cache</i> file entries.
“address class”	Set to IN for Internet
“record type”	Set to NS
“record specific data”	Contains the fully qualified host name of the name server, ending in a period

AV 8-21

Example *db.somedom.com*

```
; name [ttl] class type data

$include named.soa

thatch      IN      A      200.100.50.1
www          IN      CNAME  thatch

guam        IN      A      200.100.50.2
ftp          IN      CNAME  guam

atlantis    IN      A      200.100.50.3

gateway     IN      A      200.100.50.4
```

GNAD0821

© The Santa Cruz Operation, Inc.

AV 8-22

Example *named.soa*

```
; name [ttl] class type data

@          IN      SOA    thatch.somedom.com.  steve.somedom.com.  (
                                96031108; Serial
                                10800;      Refresh
                                1800;      Retry
                                3600000;   Expire
                                86400 );   Minimum (default ttl)

          IN      NS     thatch.somedom.com.
          IN      NS     ahost.otherdom.com.
```

GNAD0822

© The Santa Cruz Operation, Inc.

8.2.7.5 The SOA record and `$include`

Several references have been made to a *start of authority* (SOA) record kept by the name server. This record is key to how your primary and secondary name servers interact, as well as providing default *time to live* values for all records in the data files.

The SOA record:

- Is included at the top of every data file in one of two ways:
 - Directly, by including the data at the top of each data file.

This is the method the **DNS Manager** uses.

- Indirectly, by using the syntax:

`$include file`

where *file* is the name of the file containing the SOA record, typically *named.soa*.

This is the method used in the example in AV 8-21. This example references the file displayed in AV 8-22.

- Has a format described in the *named.soa(4tcp)* manual page.

Set the fields of these records as follows:

“name”	Set to the domain name
“address class”	Set to IN for Internet
“record type”	Set to SOA
“hostname”	The name of the authoritative host for this domain
“e-mail address”	The e-mail address of the person responsible for this domain. The “@” in the address is replaced by a “.”.
“Refresh”	The interval in seconds that secondary name service waits between checking for changes.
“Transfer retry”	If a zone transfer fails the length of time, in seconds, the secondary waits before trying again.
“Expire”	The maximum time in seconds that a secondary name server uses data without it being refreshed.
“Time-to-live”	The default time in seconds that a record’s data remains valid in a cache without being refreshed. This can be overridden by the records time-to-live value (ttl).

The only field that needs further explanation is the serial number. This field:

- Must be updated every time changes are made to the primary server’s data files
- Should not contain a decimal point, because it breaks other name servers not made by SCO

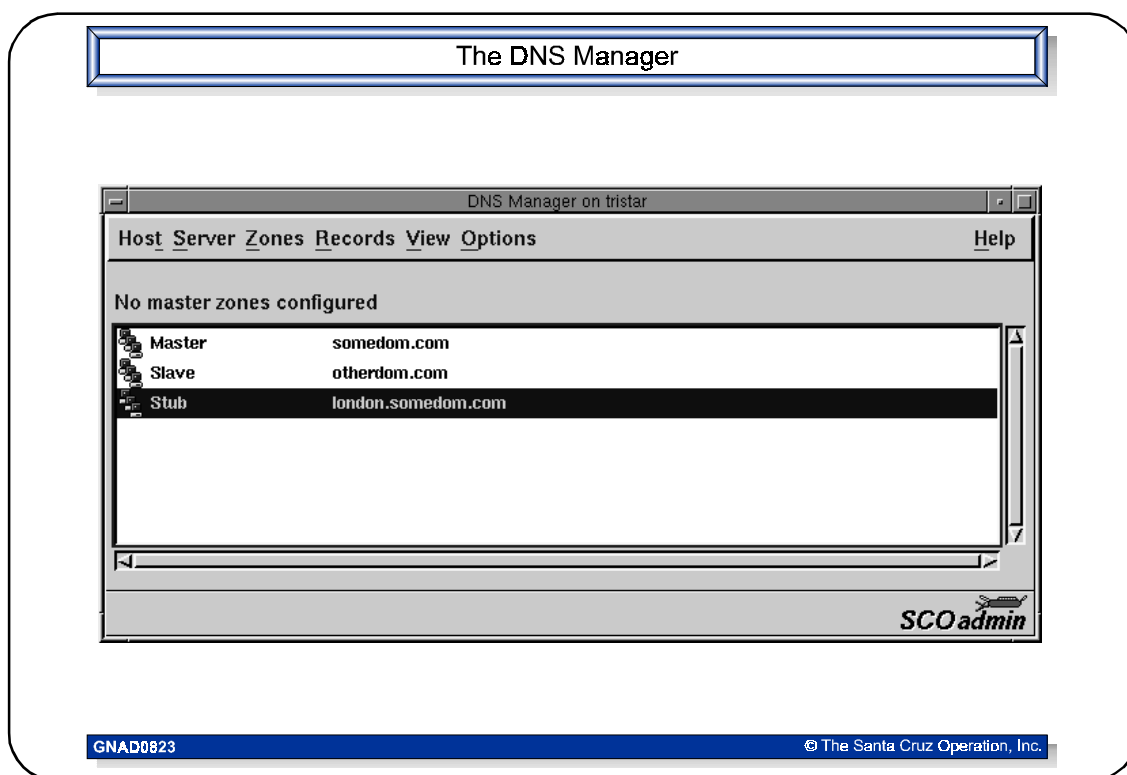
- Must increase from one version to the next. This is because the SNSs need to know they are getting *new* information.

An easy way to ensure that each number is greater than the last is to set the serial number to *YYMMDDHHX*, where:

- *YY* is the year of the change (use *1YY* after the year 1999)
- *MM* is the month of the change
- *DD* is the day of the change
- *HH* is the hour of the change
- *X* is a counter for changes that hour

This method also has the benefit of indicating when the last data was last changed.

AV 8-23



8.2.8 The DNS Manager

The **scoadmin(1M) DNS Manager** is a convenient tool for setting up and maintaining the */etc/inet/named.conf* file and the files in the */etc/inet/named.d* directory necessary for configuring name service.

Start the **DNS Manager** by running **scoadmin(1M)** and selecting **Networking** ⇨ **DNS Manager**.

The **DNS Manager** can:

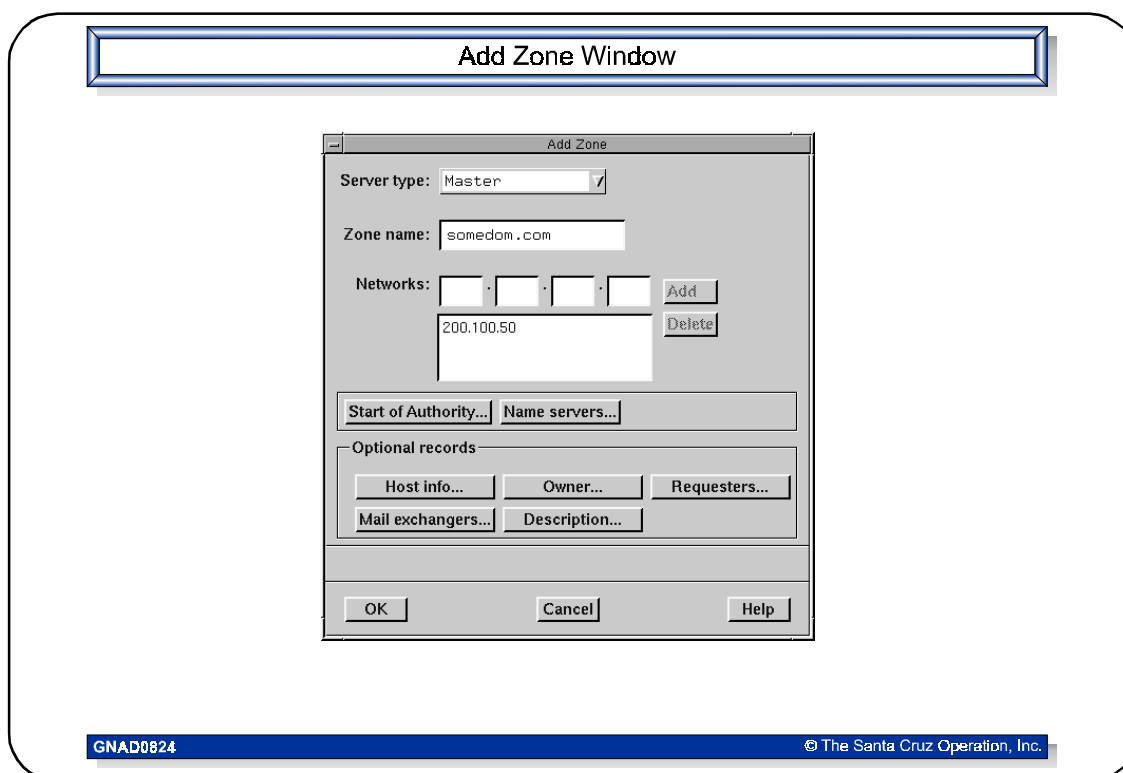
- Display information about the zones currently configured for name service
- Allow the server to be configured as a caching-only server
- Configure the addition or deletion of zones to be served as either a primary, secondary, or stub server
- Modify existing zones, e.g. to add or delete records
- Set general name service parameters such as forwarders
- Control the **named** daemon
- Define default resource records for zones yet to be configured

The **DNS Manager** has two views that can be selected from the **View** menu:

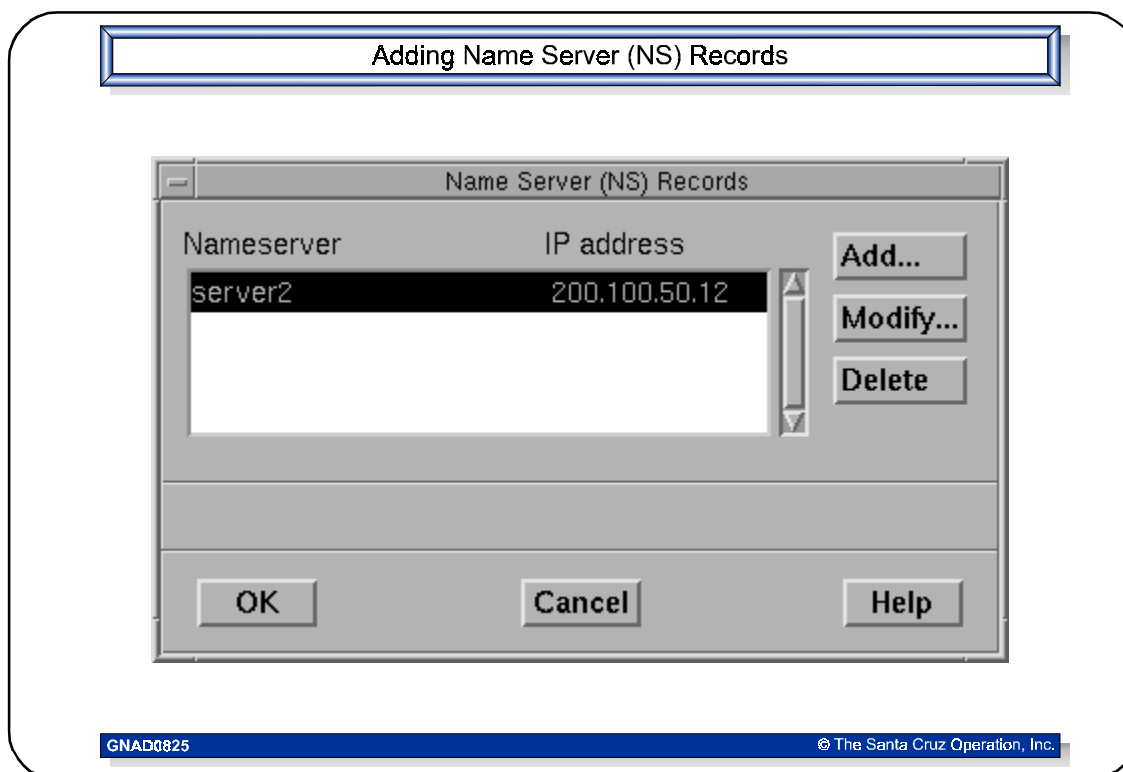
- A **Zone** view displays the configured zones and allows you to add, delete, or modify zones.
- A **Records** view displays the records of the selected zone and allows you to add, delete, or modify records for that zone.

Note: If you configure your system as a name server, you probably also want to configure it as a client to use itself as a server. This does not happen by default.

AV 8-24



AV 8-25



8.2.8.1 Configuring a name server

After starting the **DNS Manager**, you can configure a name server for a new zone by following these steps:

- Step 1. Select **Zones** ⇒ **Add** to display the **Add Zone** window.
- Step 2. Select the type of server you want to configure: Primary, Secondary, stub, or caching-only.

If you select caching-only, this is the only step necessary.

After a system is configured to be a caching-only server, to configure another type of name server on the same system requires first deleting the caching-only configuration.

Note: If you exit the **DNS Manager** with no zones configured, it will ask if you want to set up a caching-only name server.

- Step 3. Enter the **Zone name**. This is the name of the domain this server is responsible for, e.g. *somedom.com*. This is typically a second-level domain but might also specify servers for lower-level domains.

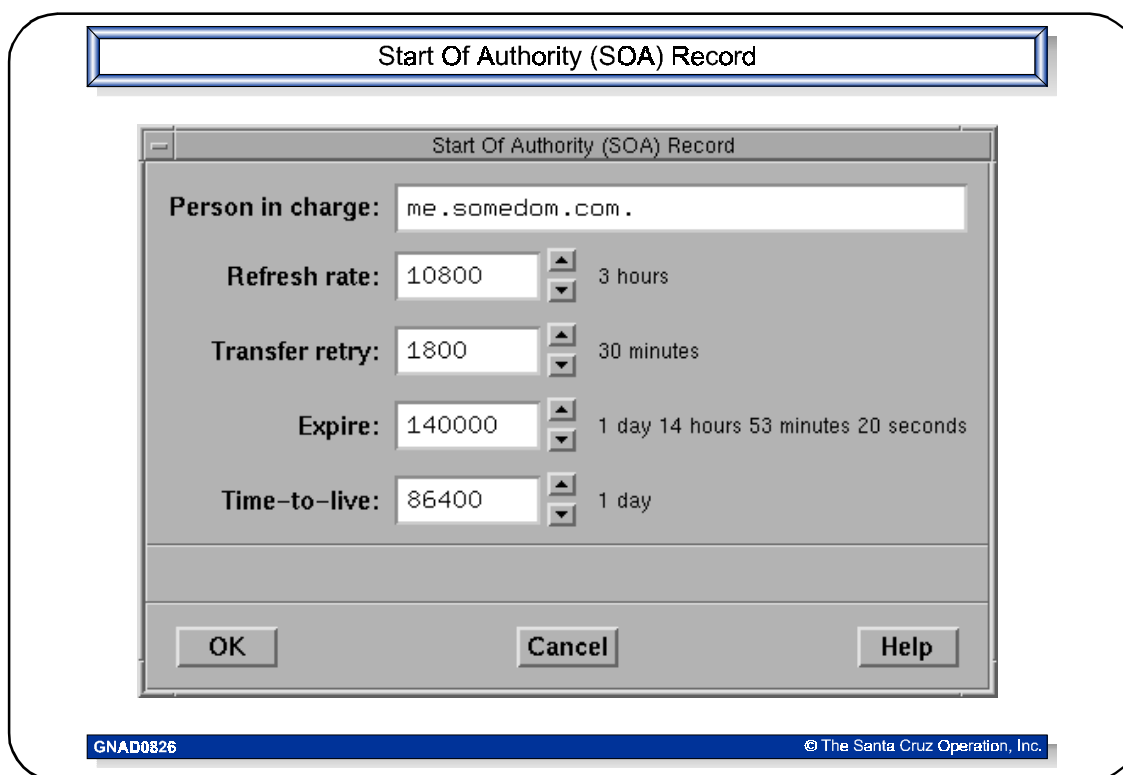
- Step 4. Enter the IP addresses of the “Networks” this zone covers. These IP addresses specify which *in-addr.arpa* domains this server has authority for.

When you add a host to this zone that belongs to these networks, the **DNS Manager** also creates the *in-addr.arpa* entries (PTR records).

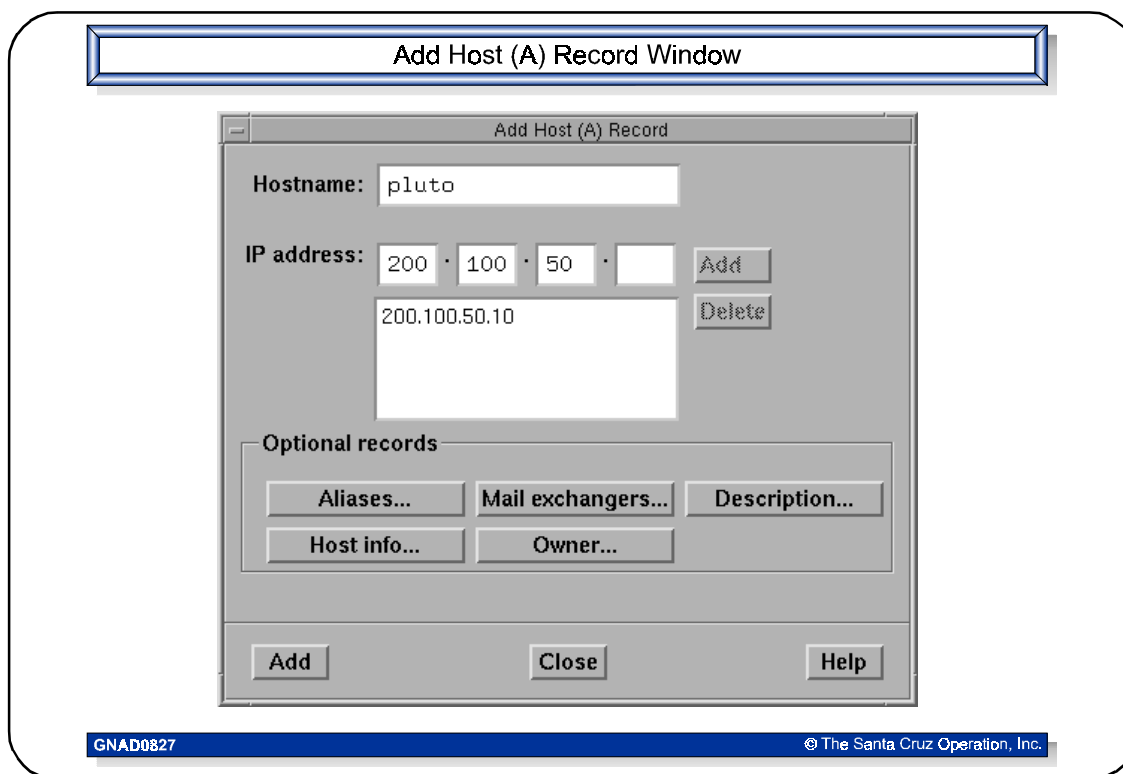
When you add a host to this zone that does not belong to these networks, the **DNS Manager** will not create the *in-addr.arpa* entries (PTR records).

Note: A minimum of one network address needs to be entered.

AV 8-26



AV 8-27



Step 5. Click on **Name servers**. This starts the Name Server (NS) Records window. You can list other primary, and secondary name servers for a zone. Enter the hostname and IP address for each name server.

This completes the configuration for secondary and stub servers. Only primary name servers require additional configuration.

Step 6. Click on **Start of Authority** to edit the start of authority record for this zone. For more information on these fields, see Section 8.2.7.5, “The SOA record and \$include,” on page 8-47.

Step 7. Click on **OK** to complete the configuration.

8.2.8.2 Adding records to a zone

Records can only be added to the primary name server, other name servers get their information from the primary. To add records for a zone on a primary name server:

Step 1. Select the zone to add the record to, then select **View** ⇒ **Records**.

Step 2. Add host records in one of two ways:

- Import a file containing host records in the */etc/hosts* file format by selecting **Records** ⇒ **Add** ⇒ **Importing file**. Only records belonging to one of the zone’s defined networks are added.

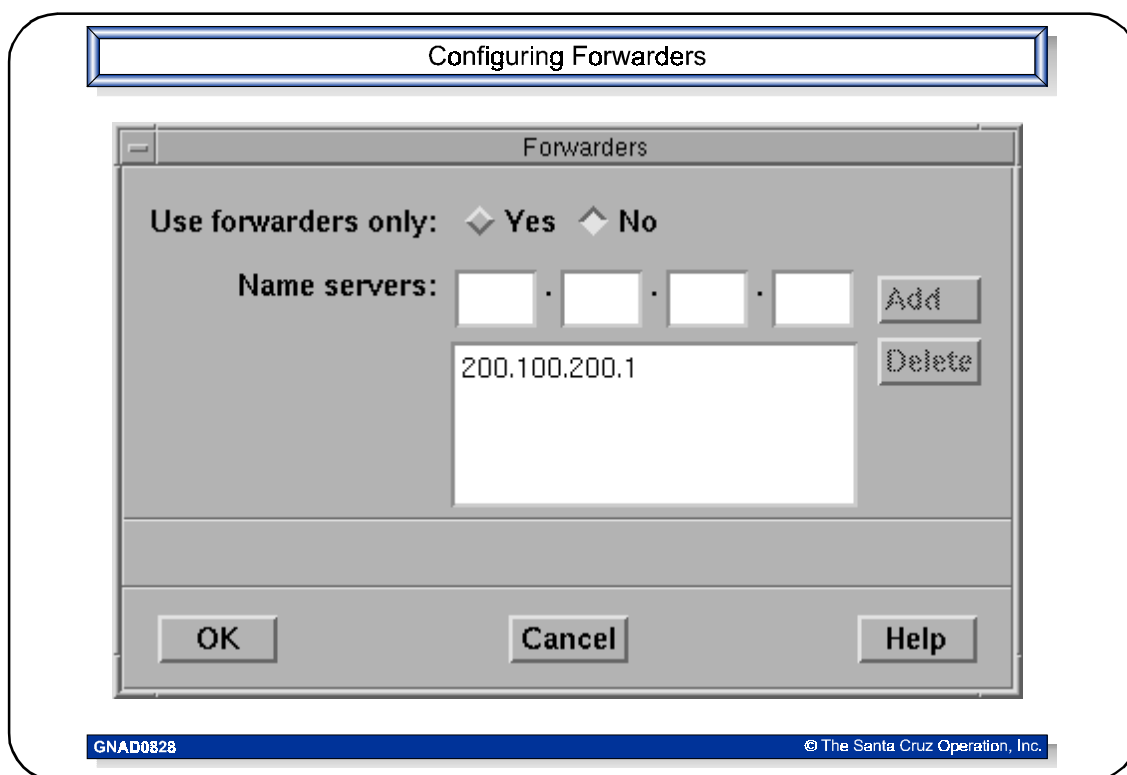
In this case, steps 3 and 4 are skipped.

- Enter records individually by **Records** ⇒ **Add** ⇒ **Host**. Enter the hostname and IP addresses for each host name.

Step 3. Use the **Aliases** button to configure CNAME records for a hostname. This allows you to attach multiple names to the same host.

Step 4. Click on **Add** to complete the configuration.

AV 8-28



8.2.8.3 Configuring server options

The **DNS Manager** can also be used to set preferences that:

- Affect the operation of the **named**(1Mtcp) server
- Apply to all domains and name servers configured on the system
- Are found under the **Server** menu:
 - **Forwarders**—Specifies the names of servers to forward requests to. Forwarders were discussed in Section 8.2.2.6, “The **forwarders** option.”

Selecting **Yes** for “Use forwarders only” makes the server only use the designated forwarders, and never contact root name servers. This is equivalent to specifying the **forward-only** option, as discussed in Section 8.2.2.7 on page 8-31.

- **Preferred**—Sets the order for sorting responses for hostnames that correspond to multiple IP addresses. This allows you to specify that certain networks get precedence over others when returning IP addresses.
- **Bad servers**—Specifies IP addresses of name servers that are not to be used.
- **Requesters**—Specifies the hosts or networks that are allowed to request zone transfers.
- **Limits**—Allows configuration of these limits on the operation of the name server:
 - **Transfer limit**—Specifies the number of simultaneous zone transfers allowed. (The default is ten.)
 - **Transfer per name server**—Specifies the number of simultaneous zone transfers allowed by one secondary name server. (The default is two.)
 - **Data segment size**—Specifies the maximum amount of virtual memory the name server can allocate to store data. (The default is 64MB.)
- **Options**—Configures the advanced server options: Recursion, Fetch glue, and Query log.

8.2.9 Controlling the name server.

The **DNS Manager** can be used to control the name server through the **Options** menu. You can:

- Initiate a zone transfer. Select **Options** ⇨ **Zone Transfer**.
- Update the root domain server cache. Select **Options** ⇨ **Update cache file**. This should be done on a regular basis to ensure the server always has an up to date root cache file. The next time the server is restarted it will re-read the cache file.
- Stop the name server. Select **Options** ⇨ **Disable server**.
- Start the name server. Select **Options** ⇨ **Enable server**.
- Test name resolution. Select **Options** ⇨ **Test configuration**. You can enter hostnames or IP addresses to see how the current name server configuration resolves them.

8.2.10 Command line control of the name server

You can also start **named**(1Mtcp) from the command line as **root**. If you start **named** with the **-d** flag, **named** places diagnostic information in the `/var/tmp/named.run` file. This diagnostic information can be very useful for troubleshooting configuration problems.

When **named** starts, it stores its process ID number in the `/etc/inet/named.pid` file. Signals can be sent to the name server to affect its operation using this PID number and the **kill**(1) command. For example to signal the name server to reread its configuration files, enter:

```
kill -HUP `cat /etc/named.pid`
```

This can be useful if you make direct modifications of the configuration files and want the name server to immediately reflect this change. For other signals, you can send the name server, refer to the **named**(1Mtcp) manual page.

AV 8-29

Example nslookup Session

```
Default Server:  thatch.somedom.com
Address:  200.100.50.1

> student4
Server:  thatch.somedom.com
Address:  200.100.50.1

Name:      student4.somedom.com
Address:  200.100.50.4
> unknown
Server:  thatch.somedom.com
Address:  200.100.50.1

*** thatch.somedom.com can't find unknown: Non-existent host/domain
> set type=PTR
> 4.50.100.200.in-addr.arpa
Server:  thatch.somedom.com
Address:  200.100.50.1

4.50.100.200.in-addr.arpa      name = student4.somedom.com
>
```

GNAD0829

© The Santa Cruz Operation, Inc.

8.3 Querying a name server

After you have set up a name server, verify that it works. Two utilities useful for testing the name server are **nslookup**(1Mtcp) and **dig**(1Mtcp).

8.3.1 The nslookup(1Mtcp) command

The **nslookup** command is well documented in the **nslookup**(1Mtcp) manual page. There are two ways to use **nslookup**:

1. Enter a series of **nslookup** commands complete with arguments from the command line. For example, to test that the *guam* entry on the example network is correct, enter:

```
nslookup guam
```

You should see the server name and its IP address as well as *guam*'s name and IP address.

2. If there are problems contacting the server or the information is incorrect, you might want to start **nslookup** in interactive mode by entering:

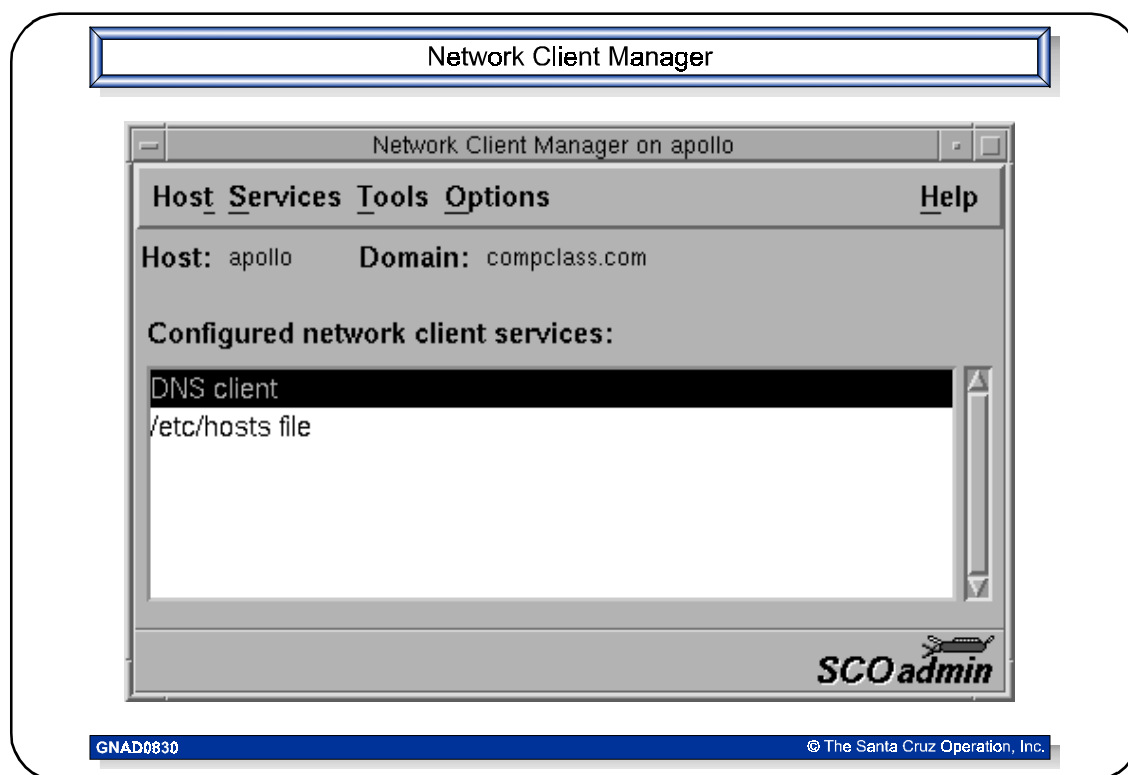
```
nslookup
```

You should receive output similar to that shown in AV 8-29. You can now issue many commands to **nslookup**. To view a list of commands, enter:

```
?
```

This mode can be very useful if you are configuring the data files and having **named** re-read its files in one window (or multi-screen) and running **nslookup** in another. You can exit **nslookup** by entering the end-of-file character, **CTRL** D.

AV 8-30



8.4 Configuring DNS Clients

After the name server(s) are configured, you can set up hosts on your network as DNS clients using the **Client Manager**.

Warning! Attempting to edit the `/etc/resolv.conf` file directly can result in the **Client Manager** being unusable.

8.4.1 DNS client configuration

To configure a DNS client from the **Client Manager**:

- Step 1. Start the **DNS Client Service Configuration**; depending on your current configuration, this is done in one of the following ways:
- Double-click on **DNS Client** in the **Client Manager**.
 - Select the **DNS Client** from the listed services and then select **Services** ⇒ **Modify**.
 - Select **Services** ⇒ **Add** ⇒ **DNS** if this service was never previously configured.
- Step 2. In the “Local domain:” field, enter the domain name to which the host belongs.
- The local domain will be used for resolving unqualified hostnames (i.e., hostnames that do not contain a dot) if the “Domain search order” field is left blank in the next step.
- Step 3. In the “Domain search order” area, specify which domains are searched and the search order for resolving hostnames that are not fully qualified:
- Enter up to six domain names.
 - Include the local domain name, if appropriate (usually included).

AV 8-31

Example *resolv.conf*

```
domain somedom.com

; query local server first, then secondary

nameserver 200.100.50.1
nameserver 200.150.75.1
```

GNAD0831

© The Santa Cruz Operation, Inc.

- Use **Add** and **Delete** to add or remove items from this list.
- Use **Promote** and **Demote** to change the search order.

If no entries are specified, the local domain name is searched.

Step 4. In the “Nameserver search order” area, specify the IP addresses of the nameservers to use when resolving a name and set the search order:

- Enter up to three nameservers.
- Use **Add** and **Delete** to add or remove items from this list.
- Use **Promote** and **Demote** to change the order of items in the list.

If the host you are configuring is also a nameserver, configure it at the top of the list by specifying the **loopback** address (127.0.0.1).

Step 5. Click on **OK** to complete the configuration.

AV 8-30 shows an example */etc/resolv.conf* file is created for any of the clients in the example network. If you set up a secondary name server within the *somedom.com* domain, then you might have half of the clients query the PNS first, then the SNS (in case the PNS failed) and the other half set up to do just the opposite.

There are three other entries covered by the documentation that are not in this example. You would only include:

- `search` and `options` if you have set up a subdomain
- `sortlist` when you are using subnets

8.5 Lesson Summary

Name servers offer you a great aid in administering your network as it grows beyond more than a few hosts or when you put your enterprise network on the Internet under your own registered domain name.

Every zone or domain served by a name server must have one primary and should have at least one secondary name server. The primary name servers contain the real data. Secondary and caching servers get the data at startup time and on demand, respectively. Clients need only be pointed at a functioning name server to query all its data.

Name servers can even forward queries for information they are not responsible for to other name servers. Queries are often passed to the root servers that can obtain the IP address of every registered name server on the Internet. In this way, the names and IP address management of the entire Internet is made possible.

Review questions

1. How many primary name servers must each registered domain on the Internet have?
2. Why is it wise to have a secondary name server outside your enterprise network?
3. Why would you configure your machine as a caching-only name server instead of a client?
4. What are A records and in which files are they found?

5. What are CNAME records and in which files are they found?

6. What are PTR records and in which files are they found?

7. What file must be created on the client hosts to point the resolver at a name server(s)?

8. What command can be used to query a name server?

8.6 Lab: Configuring and Testing Name Servers and Clients

The goal of this lab is to have you set up one or more name servers for your classroom network. Your instructor will provide you with all of the relevant information including:

- Which host will be the primary name server
- Which hosts, if any, will be secondary or caching-only servers
- Which machines will be clients

This lab consists of three parts:

- Configuring the name servers
- Testing the work completed in the first part
- Setting up and test the clients

8.6.1 Configuring servers

Your instructor will provide you with the relevant data needed to create a primary, and perhaps secondary and caching-only servers:

- If your host is to be a primary, secondary or caching-only server, follow the procedure in the next section.
- If your host is to be a client, team up with one of the students to set up their server.

Procedure for name servers

- Step 1. Run **scoadmin(1M)**. Select **Networking** ⇒ **DNS Manager** to start the **DNS Manager**.
- Step 2. Select **Zones** ⇒ **Add** to display the Add Zone window.
- Step 3. Select the type of server you want to add from the “Server type” field.
- Skip to Step 9 to complete the configuration if you are configuring a caching-only server.
- Step 4. Enter the “Zone name” for the domain to be served.
- Step 5. Enter the IP addresses for the “Networks” belonging to the zone.
- Step 6. Click on **Name servers** and add the hostnames and IP addresses for the other name servers of this domain.
- Your hosts name should appear in this list by default. Do not remove it.
- Step 7. Click on **OK** when all the other name servers have been added to return to the Add Zone window.
- Skip to Step 9 to complete the configuration if you are configuring a secondary or stub server.
- Step 8. Click on **Start of Authority**. Enter an e-mail address (replacing the “@” with a “.”) and appropriate values for the time fields.
- Step 9. Click on **OK** when finished to return to the Add Zone window.
- Step 10. Click on **OK** to complete the configuration.

Procedure for adding records to a primary server

- Step 1. Run **scoadmin**. Select **Networking** ⇒ **DNS Manager** to start the **DNS Manager**.
- Step 2. Select the zone to which you are adding records. This server must be a primary name server for that zone.
- Step 3. Select **View** ⇒ **Records**.
- Step 4. Select **Records** ⇒ **Add** ⇒ **Host** to display the Add Host (A) Record window.
- Step 5. Enter the “Hostname” of the host you are adding.
- Step 6. Enter the “IP Address” of the host you are adding.
- Step 7. Click on **Add** to the right of the IP address field.
- Step 8. Click on **Aliases** if this host will have any alias names.

If this host does not have any aliases, skip to Step 12.
- Step 9. Enter the alias name in the “Aliases” field.
- Step 10. Click on **Add**.
- Step 11. Click on **OK** after all the aliases have been added.
- Step 12. Click on **Add** to add the record.
- Step 13. Repeat Steps 5 through 12 to add more records.
- Step 14. Click on **Close** when you have finished adding records.

8.6.2 Testing your server

The objective of this lab is for you to become familiar with utilities to test and query name servers. Working on the same host on which you configured a name server, verify that you can retrieve both the A and PTR records using either **dig** or **nslookup**.

Procedure

- Step 1. Make sure you are still logged in as **root**.
- Step 2. Use **nslookup** from command mode to look up both the name and IP address of at least two hosts on your network.
- Step 3. Use **nslookup** in interactive mode. The query type defaults to A records. Check the IP addresses of at least two hosts on your network by entering their names.

Change the query type to PTR and look up the names of at least two hosts on your network by entering in their reverse IP addresses in the *in-addr.arpa* domain.

8.6.3 Configuring a client

The objective of this lab is for you to configure a client host to query multiple name servers in case one is down, and to fall back on the `/etc/hosts` file if neither server can be found.

Procedure

- Step 1. Log in as **root**.
- Step 2. Enter the **Client Manager** tool from the **Networking** folder.
- Step 3. Select DNS by double clicking on “DNS client” in the **Client Manager** or if it is not there by selecting **Services** ⇒ **Add** ⇒ **DNS**.
- Step 4. Enter the local domain name provided by the instructor in the “Local domain” field.
- Step 5. Enter the domain specified by the instructor in the “Domain search order” area.
- Step 6. Click on **Add** in this area to submit your entry to the domain search order list.
- Step 7. Repeat Steps 5 and 6 for any other domains the instructor provided.
- Step 8. Enter the name server IP address provided by the instructor in the “Nameserver search order” area.
- Step 9. Click on **Add** in this area to submit your entry to the domain search order list.
- Step 10. Repeat Steps 5 and 6 for any other name server IP addresses the instructor provided.
- Step 11. Check the nameserver resolution order by selecting **Services** ⇒ **Name Resolution Order** and change if necessary.

Procedure for testing client configuration

- Step 1. Use **nslookup** to query for the IP address of one or more of the machines on your LAN.
- Step 2. Stop the name server on the host that you listed as your first choice in your */etc/resolv.conf* file.
- Step 3. Use **nslookup** again to query for IP addresses. You might experience delays as the resolver times out on the first server and then queries the second server.
- Step 4. Stop the name server on the host that you listed as your second choice in your */etc/resolv.conf* file.
- Step 5. Try using a client such as **telnet** to contact a host listed in your */etc/hosts* file. Again, you might experience delays as the resolver times out after trying each of the name servers and then queries the */etc/hosts* file.

8.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

named (1Mtcp)	The name server daemon
nslookup (1Mtcp)	Used to query name servers
scoadmin (1M)	Menu-driver program for configuring the UnixWare 7 system. It includes the DNS Manager and the Client Manager .

Relevant files

<i>/etc/inet/named.boot</i>	Is the default file read by named . Refer to the <i>named.boot</i> (4tcp) manual page.
<i>/etc/inet/named.d</i>	Is the default directory for locating named database files in UnixWare 7.
<i>/etc/named.d</i>	Is the default directory for locating named database files in the SCO OpenServer Release 5 operating system.
<i>/etc/resolv.conf</i>	Contains the <i>domain</i> and <i>nameserver</i> entries to point the client at name server. Refer to the <i>resolv.conf</i> (4tcp) manual page.
<i>/usr/lib/named</i>	Is the default directory for locating named database files in SCO operating systems previous to SCO OpenServer Release 5.
<i>db.somedom.com</i>	Is the example name for the file that contains information about hosts in a zone when lookups are being done using a host's name. Refer to the <i>db.zone</i> (4tcp) manual page.
<i>db.cache</i>	Is the default name for the file containing the A and NS records for the root name servers. Refer to the <i>db.cache</i> (4tcp) manual page.
<i>db.local</i>	Is the default name of the file containing the PTR record for <i>localhost</i> . Refer to the <i>db.local</i> (4tcp) manual page.

<i>db.50.100.200</i>	Is the example name of the file that contains information about hosts in a zone when lookups are being done using a host's IP address. Refer to the <i>db.reverse_address(4tcp)</i> manual page.
<i>named.soa</i>	Is the default name for the file containing the SOA record and NS records for a zone. Refer to the <i>named.soa(4tcp)</i> manual page.

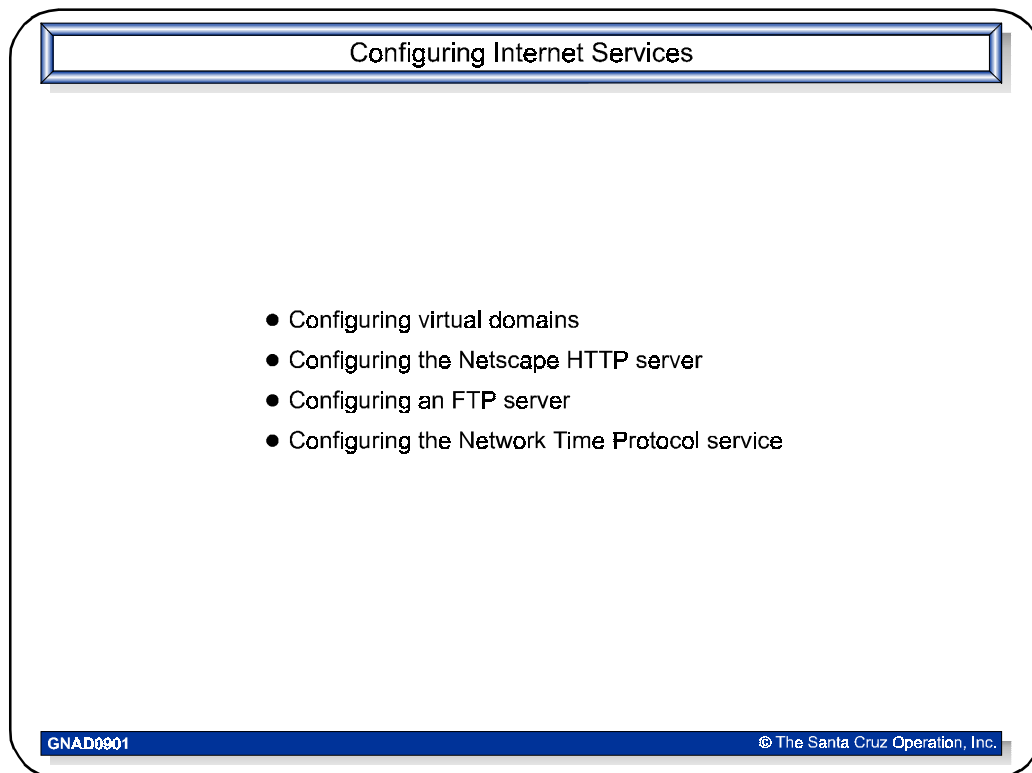
8.8 Answers to Review Questions

1. One primary name server is required for each registered domain on the Internet. If a domain is divided into several zones, each zone must have its own primary name server.
2. If you have a secondary name server outside your enterprise network (for example, at your ISP), then if your enterprise network should go off the Internet for any reason, the addresses of machines, and more importantly, mail exchangers for your enterprise network, can still be looked up by others on the Internet.
3. Configuring the machine as a caching-only name server instead of a client improves response time for recently requested names and reduces network traffic.
4. A records map host names to IP addresses. They are found in the host database files such as *db.somedom.com*.
5. CNAME records allow you to create nicknames or aliases for hosts. They are found in the host database files such as *db.somedom.com*.
6. PTR records map IP addresses to names. They are typically found in the reverse lookup, *in-addr.arpa* database files such as *db.50.100.200*.
7. The */etc/resolv.conf* file must be created on all client machines to point the resolver at a name server(s).
8. The **nslookup**(1Mtcp) command can be used to query a name server.

9

Configuring Internet Services

AV 9-1



Lesson 9

Configuring Internet Services

Introduction

The previous lessons have discussed how a TCP/IP protocol stack is configured, how data is transmitted, and how names and other attributes of machines are looked up using machines running TCP/IP in a LAN or WAN or both.

UnixWare 7 provides several Internet-standard services that make a network more usable. Understanding what these services are and how to configure them will enhance the value of your companies network and make your job as a network administrator easier.

Objectives

This lesson describes the Internet-standard and other network services available on UnixWare 7 and how to configure them. By the end of this lesson, you will be able to:

- Create a virtual domain and configure virtual domain services
- Configure an HTTP server
- Configure an FTP normal and anonymous FTP server
- Configure the Network Time Protocol (NTP) server and clients

9.1 Virtual Domains

By default, your system has exactly one IP address associated with each network interface or PPP connection, and exactly one system name. However, it is sometimes useful to have the services on the system behave differently, depending on which network interface is used to access it. For example, if you have two network interfaces on your system, one that connects to your LAN and one that connects to the Internet, you might want the World Wide Web server on your system to provide one set of information to people internal to your organization, and another set of information to everyone else.

Some sites, particularly ISPs, want to take this concept one step further. By using IP aliasing it is possible to make a single network interface respond to multiple IP addresses. This allows a single system to be known by multiple names. A single system can serve multiple World Wide Web sites and nobody accessing these web sites can tell that these sites are being served by a single machine.

This process of using a single system to respond to multiple IP addresses and system names is called *multi-homing*. The multiple distinct environments that the system presents to the outside world are called *virtual domains*.

UnixWare 7 can support up to 254 virtual domains, each with distinct Internet services. This lesson describes how to configure multiple virtual domains to provide World Wide Web, POP (Post Office Protocol) e-mail, and FTP services for your UnixWare 7 system.

In reality, there is no such thing as a virtual domain, only virtual hosts. However, because all literature, online documentation, and tools use the term virtual domain, this course also uses the term. It might help to think of a virtual domain as a virtual host residing on your computer.

AV 9-2

Example S70ipaliases File

```
#!/sbin/sh
echo "setting up IP aliases:"
echo "\t150.200.76.99"
ifconfig net0 alias 150.200.76.99
```

GNAD0902

© The Santa Cruz Operation, Inc.

AV 9-3

Example Additions to /etc/inet/config

```
4a:/usr/sbin/in.gated::Y:/etc/inet/gated.conf::
# The next line can be read:
# If field 3 is a Y and /usr/sbin/in.gated was not run, then run
# /usr/sbin/in.routed
4b:/usr/sbin/in.routed:/usr/sbin/in.gated:Y:::
#4c:/usr/sbin/route::n::add default router_placeholder:
6:/usr/sbin/in.xntpd::y:/etc/inet/ntp.conf::
7:/usr/bin/sh::y:/etc/inet/rc.inet:/etc/inet/rc.inet start:
# By default start address allocation and dhcp servers
8a:/usr/sbin/in.aasd::y:/etc/inet/aasd.conf::
#8b:/usr/sbin/in.dhcpd::y:/etc/inet/dhcpd.conf:-t 0:
# Entry to start bootpd, comment-out dhcpd line
#9a:/usr/sbin/in.bootpd::y:/etc/inet/bootptab:-t 0:
# Entries to start both bootpd and dhcpd, comment out dhcpd entry
#9b:/usr/sbin/in.bootpd::y:/etc/inet/bootptab:-t 0 -S -P bootps-alt:
#9c:/usr/sbin/in.dhcpd::y:/etc/inet/dhcpd.conf:-t 0 -b bootps-alt:
4c:/usr/sbin/route::y::add default 198.140.193.1:
4d:/usr/sbin/route::y::add 224.0.0.0 198.140.193.111:
4e:/usr/sbin/ifconfig::y::net0 alias 150.200.76.99:
```

GNAD0903

© The Santa Cruz Operation, Inc.

9.1.1 Creating virtual domains

To create a virtual domain:

- Step 1. Alias an IP address to an existing network interface (other than the loopback) using the **ifconfig**(1Mtcp) command. For example, to add the IP address *150.200.76.30* to the **net0** network adapter, enter:

```
ifconfig net0 alias 150.200.76.30 netmask 255.255.255.0
```

The **ifconfig** commands only persist until the system reboots. To make the alias permanent, either:

- Create the */etc/rc2.d/S70ipaliases* file with the **inconfig** entries. This file does not exist by default. You must create a valid shell script with appropriate permissions. This is shown in AV 9-2.
- Enter the commands into the */etc/inet/config* file as shown in AV 9-3.

- Step 2. Set up routing to this IP address. If the address resides within:

- The same subnet as the card's native IP address, do not configure any additional routing
- A different subnet than the card's native IP address, advertise a host route

For more information on routing, see **scohelp** ⇒ **Internet and Intranet** ⇒ **Administering TCP/IP and Internet services** ⇒ **Setting up subnets and routing** ⇒ **Configuring a system for IP routing**.

- Step 3. Set up the individual services that support virtual domains, as described in the remainder of this lesson.

Exercise: IP address aliasing

You must be logged in as **root** to complete this exercise.

Step 1. Determine an IP address you can alias to one of your network adapters. For simplicity, use an address on the same subnet as your primary LAN interface.

Step 2. Create an IP address alias. Enter:

```
ifconfig net0 alias ipaddr netmask mask_value
```

Use the IP address and netmask values supplied by your instructor.

Step 3. Test the alias. Enter:

```
ping ipaddr
```

9.1.2 Virtual domain services

After virtual domains are enabled and created, individual network services can be configured for each virtual domain.

Virtual domain support is server-daemon dependent. Currently, only four services support virtual domains:

DNS	DNS can serve names, IP addresses, and other records for virtual domains.
Web	The Netscape FastTrack Server can be configured for virtual domains just as web services are configured for physical network interfaces.
FTP	Anonymous FTP can be configured for virtual domains in a similar manner to configuring anonymous FTP for physical interfaces.

9.1.3 DNS and virtual domains

One of the important properties of a virtual domain is its name (for example, *abc.com* or *web.somedom.com*). This is the name by which services associated with a virtual domain are known—it is the *virtual system name*.

To allow other systems to access a virtual domain or physical interface by its name, create entries for it on a DNS server. These entries need to be made on the DNS server that is acting as the primary server for the domain containing the virtual domain (host).

For example, if you created a virtual domain called *web.somedom.com* with IP address *150.200.76.30* for web services, add a record for it on the master name server just as you would any new host on your network:

- Step 1. Start the **DNS Manager**.
- Step 1. Select the zone to add the record to, then select **View** ⇒ **Records**.
- Step 2. Select **Records** ⇒ **Add** ⇒ **Host**. Enter the host name and IP addresses for each host name and click on **Add**.
- Step 3. Click on **Add** at the bottom of the screen when finished entering host records.
- Step 4. Click on **Host** ⇒ **Exit** to exit the **DNS Manager**.

You can now access *web.somedom.com* from the local network using the name, rather than the IP address.

Note that if you create a virtual domain (host) with a different domain name, such as *www.otherdom.com*, you will need to manually configure DNS to create a primary name server for *otherdom.com*. This can be accomplished by editing */etc/inet/named.conf* and copying the DNS configuration records and files for an existing primary name server.

See Lesson 8: “Domain Name System (DNS) Configuration” for more about DNS. Also refer to **scohelp** ⇒ **Internet and Intranet** ⇒ **Configuring the Domain Name Service (DNS)** for more information.

9.2 Configuring and Using Web Services

The Internet has experienced explosive growth for two main reasons:

- A gigantic amount of information is available.
- The information is easily accessed using a *web browser*. Web browsers for UNIX[®] systems include Netscape Navigator™, Netscape Communicator™, Lynx, and NCSA Mosaic.

Web servers provide the information displayed by web browsers and use the HyperText Transport Protocol (HTTP) to exchange information with the web browser.

Web servers available for SCO platforms include:

- Netscape FastTrack Server™—included with most UnixWare 7 system profiles. This is the server described in this course.
- Apache—available on Skunkware. More information is available at <http://www.apache.org>.

9.2.1 Starting and stopping Netscape HTTP servers

By default, UnixWare 7 systems are only running one web server. This server runs on port 457 and serves **scohelp** and manual pages.

There are two other web servers you might want to start:

- The Netscape Administration Server. This server allows you to add, remove, and configure existing Netscape web servers. It runs on port 620.
- The Netscape FastTrack web server on port 80. This is the web server that is used by default when web browsers contact your host. All web browsers access port 80 by default.

To start this server, and have it start every time your system boots, enter:

```
nsfast enable 80
```

The **nsfast(1M)** command has the general form:

```
nsfast action [servername]
```

where:

- The *action* is one of **start**, **stop**, **enable**, **disable**, or **restart**.
 - **start**, **stop**, and **restart** are one-time actions
 - **enable** and **disable** also configure whether the server starts at boot time
- The *servername* matches the suffix of any directory under the */usr/ns-home* directory that has a prefix of *httpd-*. For example, there is a */usr/ns-home/httpd-80* directory for the default Netscape FastTrack web server and this can be started with the command:

```
nsfast start 80
```

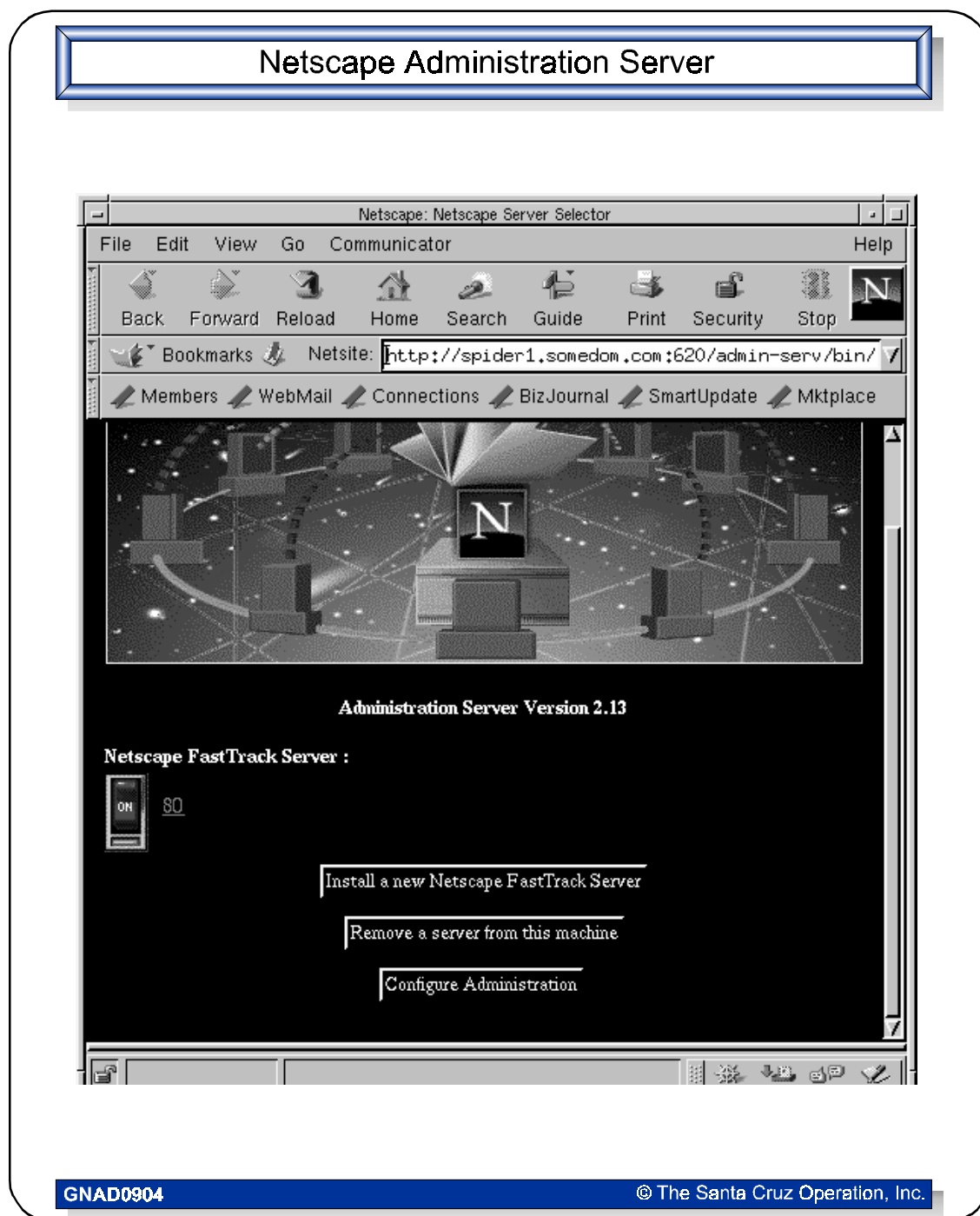

Exercise: Starting the Netscape FastTrack Server

Use this command to start the default web server:

nsfast enable 80

This web server will now start each time UnixWare 7 boots.

AV 9-4



9.2.2 Configuring the Netscape FastTrack Server

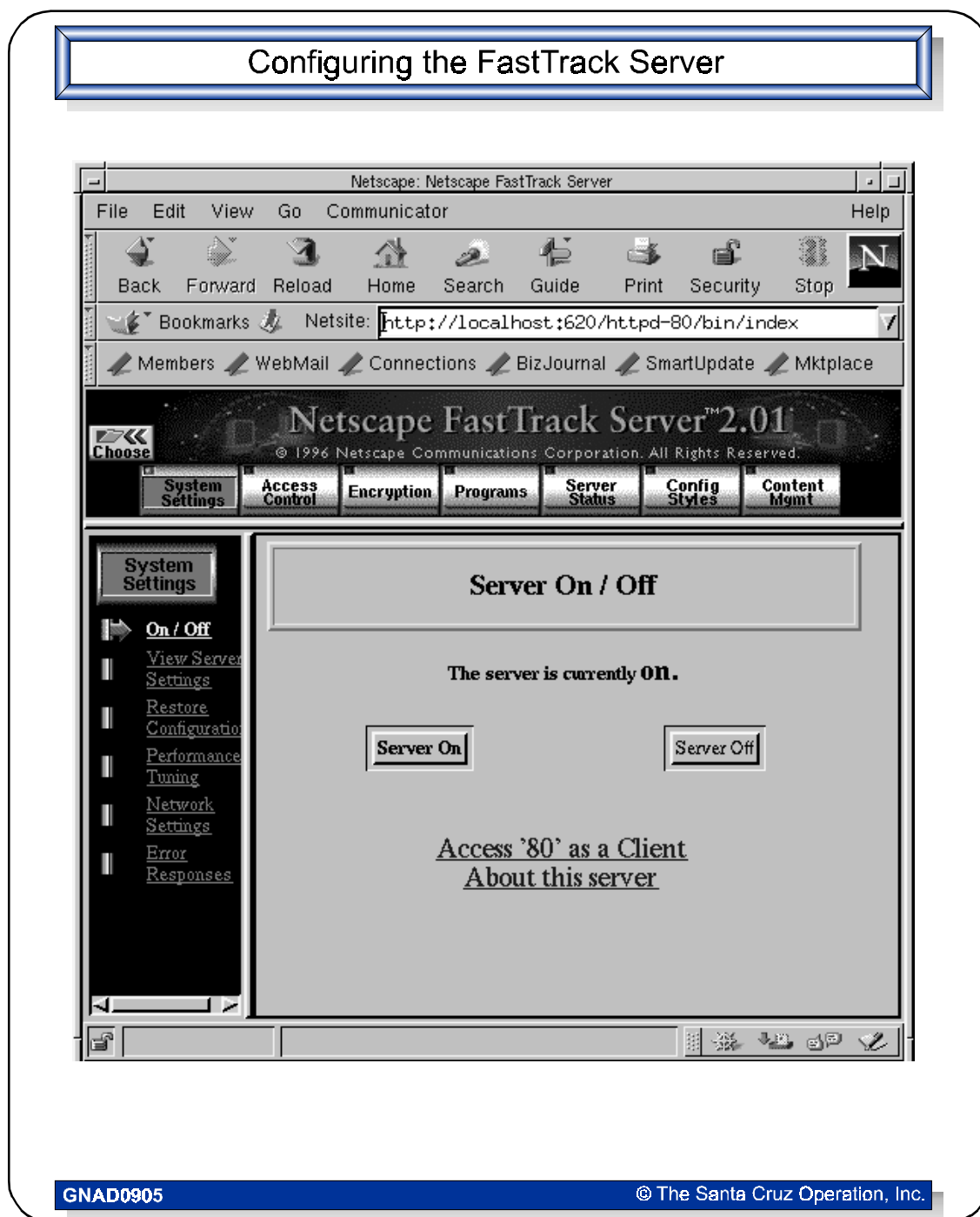
The Netscape **Administration Server** is another web server used to configure the FastTrack web server. It is started from the **Netscape Server Admin Manager** and runs on port 620. The **Administration Server** allows you to add, remove, and configure existing web servers.

From the Netscape **Administration Server**, you can configure these features of your FastTrack server:

System Settings	Sets information about the FastTrack Server including the directory location of server binaries, the host name URL clients use to access the server, and the port used to listen for HTTP requests
Access Control	Allows you to protect files and directories based on user name, password, domain name, or IP address
Encryption	Configures the Secure Sockets Layer (SSL) 3.0 protocol, enabling encrypted and authenticated transactions
Programs	Configures server-side applications, including Java applets and CGI programs, that provide extra server functionality such as games or search tools
Server Status	Provides tools for monitoring the server and using log files
Configuration Styles	Applies a set of options to files or directories maintained by your server
Content Management	Provides HTML editing and configures public information directories to let users on your system create home pages and other documents

The next sections describe how to run the Netscape **Administration Server** from the local server and from remote systems. For detailed information about configuring the Netscape FastTrack server, see **scohelp** ⇒ **Internet and Intranet** ⇒ **Netscape** ⇒ **Netscape FastTrack Server Administrator's Guide**.

AV 9-5

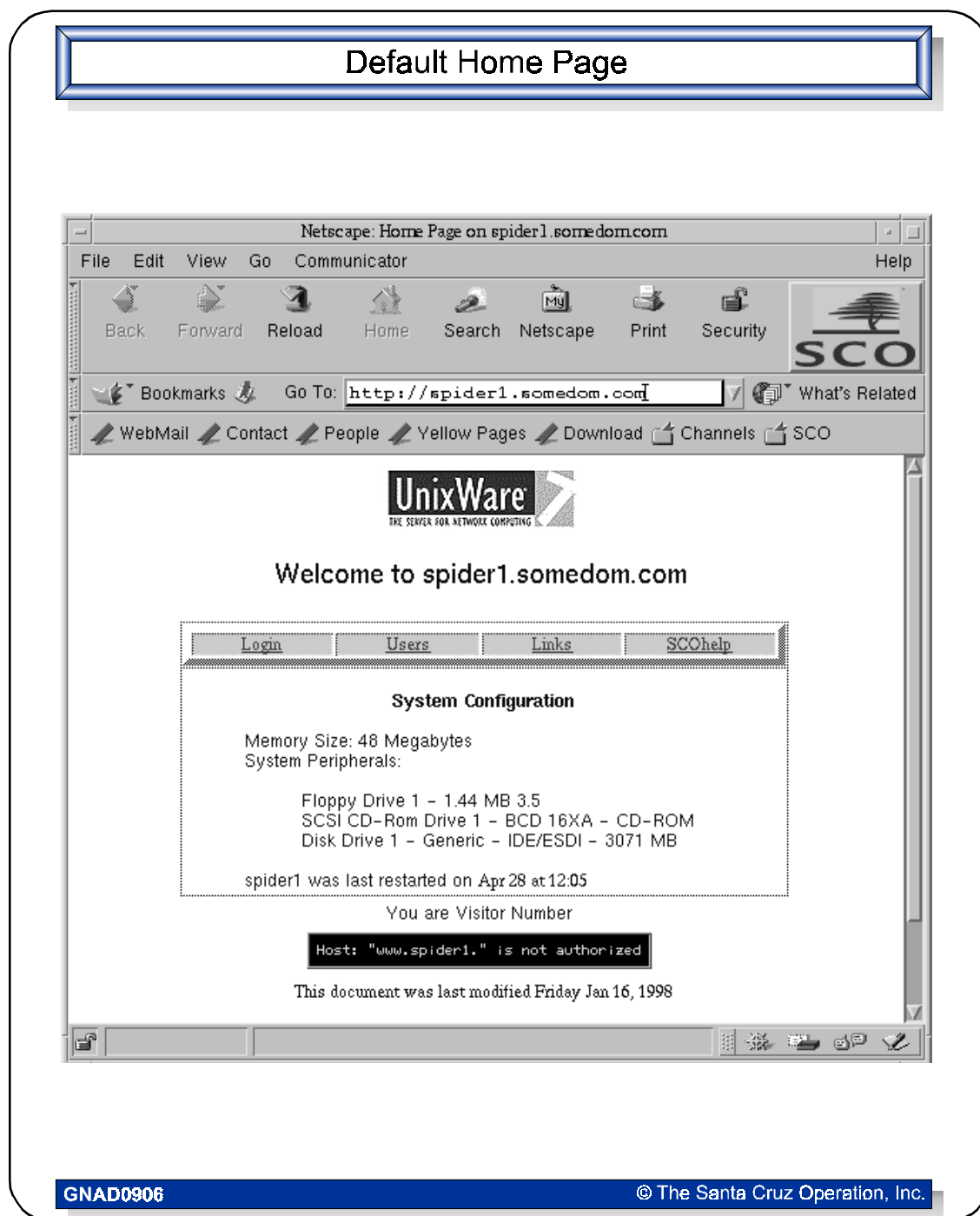


9.2.3 Running the Netscape Administration Server

To administer the Netscape FastTrack server, make sure the FastTrack server process has been started (Section 9.2.1 on page 9-12) and then access the **Administration Server** using the Netscape Communicator browser:

- Step 1. Run **scoadmin**(1M).
- Step 2. Select **Netscape Server Admin**. Enter the user name **admin** and the original password for the **root** account at the time of system installation.
- If you cannot remember the password, remove the password by editing `/usr/ns-home/admserv/admpw` and deleting everything after the colon (:) character.
- NOTE: If you have changed your system name, IP address or domain name, you will not be able to access the **Administration Server**. You must first edit `/usr/ns-home/admserv/ns-admin.conf` to reflect these changes.
- Step 3. Click on the link for the server you want to configure. If you have only one network interface configured, click on the link 80 next to the On/Off switch, as shown in AV 9-5.
- You can now view and modify the web server configuration.
- Step 4. Click on View Server Settings in the left frame.
- Step 5. Click on **Server Status** in the top button bar. Scroll down through the right frame to view the access log.
- Step 6. Click on **System Settings** in the top button bar. Click on On/Off in the left frame to return to the main configuration page.
- Step 7. Stop and restart the server whenever you make any configuration changes.
- Step 8. Reload URL <http://hostname:620> or use the **Back** button to return to the **Server Selector**.

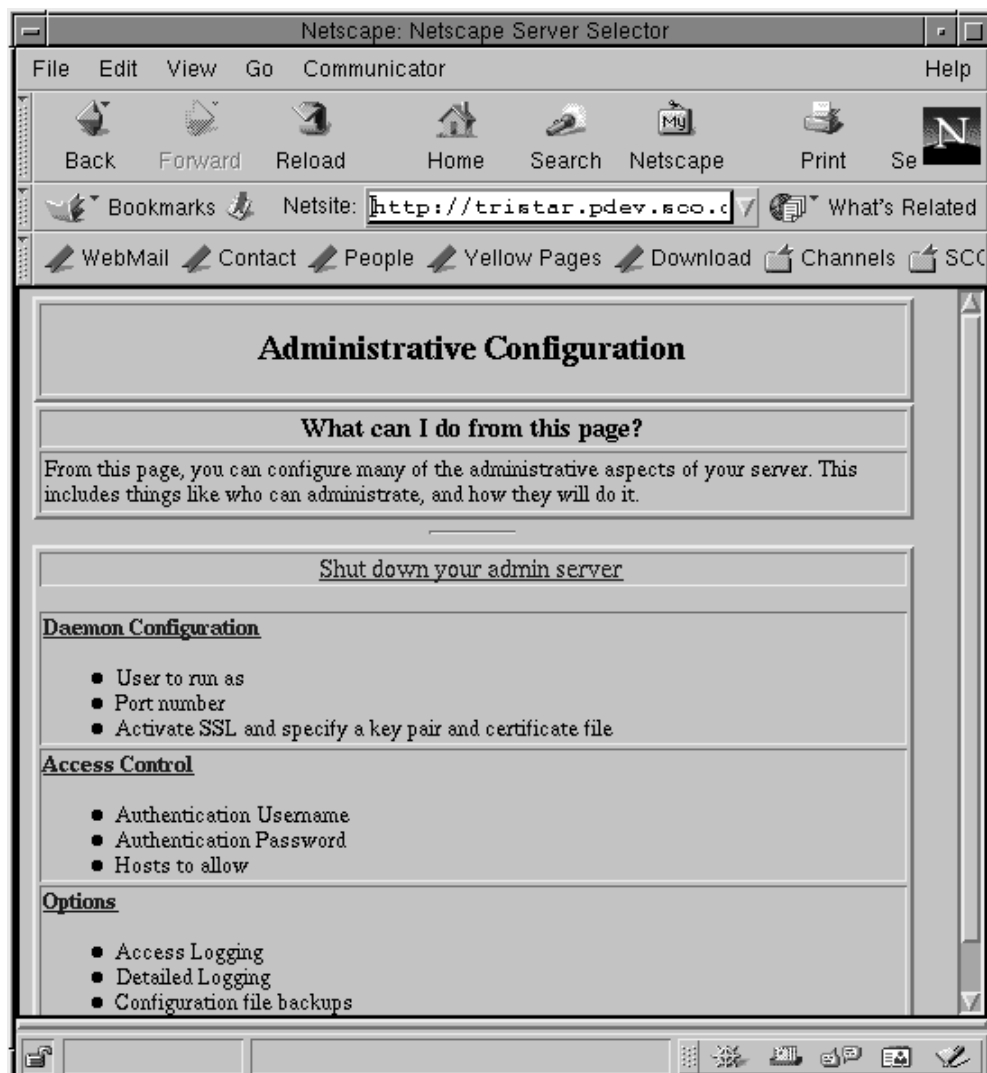
AV 9-6



Exercise: Running the Netscape Administration Server

- Step 1. Start the **Administration Server** from **scoadmin** ⇒ **Netscape Server Admin**.
- Step 2. Enter the user name **admin** and the original password for the **root** account at the time of system installation.
- Step 3. Click on the link for the server you want to configure. If you have only one network interface configured, click on the link 80 next to the On/Off switch.
- Step 4. Verify that you see the page shown in AV 9-6.
- Step 5. Click **Server On** to start the FastTrack server, if it is not already running.
- Step 6. Click on Access '80' as a Client. This opens another browser window and displays the root home page, as shown in AV 9-6.
- Step 7. Click on **Server Off**. When you see the **Success!** pop-up window, click on **OK** to confirm that the server has been shut down.
- Step 8. In the other browser window, try to access the same URL (using the **Reload** button). A Netscape error message is displayed.
- Step 9. Click **Server On** to restart the server. Click on **OK** to confirm that the server has been started.
- Step 10. Reload the URL again in the other browser window to verify that the server is running.

Netscape Server Selector-Configuring Administration



GNAD0907

© The Santa Cruz Operation, Inc.

9.2.4 Configuring remote administration

To access the Netscape **Administration Server** from any host on the network, configure *remote administration*.

The **Administration Server** is configured to only allow access from a browser running on the local machine by default. There are two ways to configure remote administration access:

- Using the **Administration Server**. This is only useful if you are on the local machine and want to configure access for other machines.
- From the command line. This is useful if you are on a remote machine and can use **telnet(1tcp)** to connect to the machine running the **Administration Server**.

9.2.4.1 Configuring remote access using the Administration Server

To configure remote administration:

- Step 1. Start the **Administration Server** from **scoadmin** ⇒ **Netscape Server Admin**.
- Step 2. Enter the user name **admin** and the original password for the **root** account at the time of system installation.
- Step 3. From the **Server Selector**, select the **Configure Administration** button.
- Step 4. Select Access Control link from the Administrative Configuration page.

Configuring Administration-Access Control

The screenshot shows a Netscape browser window titled "Netscape: Netscape Server Selector". The address bar displays "http://spider1.somedom.com:620/admin-serv/bin/". The main content area is titled "Administration Server Access Control" and contains the following fields:

- Hostnames to allow:**
- IP addresses to allow:**
- Authentication user name:**
- Authentication Password:**
- Authentication Password (again):**

At the bottom of the form are three buttons: **OK**, **Reset**, and **Help**. Below the buttons is a link: [Return to Administrative Configuration](#).

GNAD0908

© The Santa Cruz Operation, Inc.

- Step 5. Specify the remote hosts. Fill in either the:
- “Hostnames to allow” field with the name(s) of hosts allowed access
 - “IP addresses to allow” field with the IP address(es) of the hosts allowed access

If more than one name or IP address is provided, use the vertical bar (|) to separate the entries, and surround the list with parentheses. You can also use a wildcard such as the asterisk (*).

NOTE: The closing parenthesis may appear to be a vertical bar.

- Step 6. Click on **OK**. The **Success!** page lets you know that your changes have been made.

- Step 7. Click on the Shut it down link, then click on the button **Shut down the administrative server**.

Note that you will return to the main **Administration Server** page. However, you will be unable to perform any configuration tasks until you restart the **Administration Server**.

- Step 8. Restart the **Administration Server** as **root** from a terminal window. Enter:
- ```
/usr/ns-home/start-admin
```

- Step 9. Verify the changes. Access the **Administration Server** from both the local and the remote hosts.

**WARNING:** If you configure remote access incorrectly, you can prevent any access to the **Administration Server**, even from the local host. If this happens, use the procedure in the next section to edit the configuration file that controls access to the **Administration Server**, `/usr/ns-home/admserv/ns-admin.conf`.

AV 9-9

The `/usr/ns-home/admserv/ns-admin.conf` File

```
NetsiteRoot /usr/ns-home
Port 620
User root
ErrorLog /usr/ns-home/admserv/errors
AccessLog /usr/ns-home/admserv/access
PidLog /usr/ns-home/admserv/pid
ServerName spider1.
Backups 10
AdminUsers /usr/ns-home/admserv/admpw
Addresses
(127.0.0.1|150.200.76.3|150.200.76.50|150.200.76.33)
Security off
```

GNAD0909

© The Santa Cruz Operation, Inc.

### 9.2.4.2 Configuring remote access from the command line

---

To configure remote administration using the command line:

- Step 1.        Log in as **root**.
- Step 2.        Edit the `/usr/ns-home/admserv/ns-admin.conf` file:
- Modify the line that begins with “Addresses” to add the IP address(es) of any other system(s) you want to access the **Administration Server**.
  - The syntax for listing multiple addresses is:  
  
    Addresses ( *a.b.c.d|e.f.g.h* )  
  
    where the letters are replaced by valid IP octets.
- Step 3.        Restart the **Administration Server**:
- `/usr/ns-home/stop-admin`
- `/usr/ns-home/start-admin`
- Step 4.        Verify the changes. Access the **Administration Server** from all remote locations. To do this, run a web browser and load URL <http://remote-host:620>.

AV 9-10

## Adding a FastTrack Server

The screenshot shows the 'Netscape: Netscape Server Selector' window. The title bar includes 'File', 'Edit', 'View', 'Go', 'Communicator', and 'Help'. The address bar shows 'Go To: http://www.spider1.som'. The main content area contains several configuration fields:

- Server Name:** web.somedom.com
- Bind address:** 150.200.76.30
- Server Port:** 80
- Server Identifier:** web
- Server User:** nobody
- Processes:** 4
- Minimum Threads:** 4
- Maximum Threads:** 32

Below these fields are three radio button options for IP resolution:

- ☒ Always attempt to resolve IP addresses into host names.
- ☐ Attempt to resolve IP addresses only for access control and CGI.
- ☐ Never attempt to resolve IP addresses into host names.

At the bottom, the **Document Root:** is set to `/var/internet/ip/150.200.76.30/httpd/doc`. The status bar at the bottom shows '100%' and various icons.

GNAD0910

© The Santa Cruz Operation, Inc.

## 9.3 Web Services for Virtual Domains

---

The Netscape **Administration Server** allows you to add, remove, and configure web servers on both physical and virtual interfaces. Once created, configuration of each web server is performed using the Netscape **Administration Server**. The process is the same, regardless of whether it is a server for a virtual domain or a single physical host.

The next sections describe how Netscape FastTrack servers are added for virtual domains. Once created, these features can be configured for each new server.

### 9.3.1 Adding a virtual domain web server

---

Before adding a new FastTrack web server for a virtual domain, create the domain (Section 9.1.1 on page 9-7) and add it to your DNS configuration (Section 9.1.3 on page 9-10).

To add a web server for a virtual domain:

Step 1. Start the **Administration Server**. Run **scoadmin** and select **Netscape Server Admin**.

Step 2. Enter the user name **admin** and the password for the **root** account.

Step 3. Click on **Install a new Netscape FastTrack Server** at the bottom of the **Server Selector** screen.

Step 4. Complete the server installation form, focusing on these fields (the other fields can use default values):

|             |                                                                             |
|-------------|-----------------------------------------------------------------------------|
| Server Name | Enter the fully-qualified domain name of an existing virtual domain (host). |
|-------------|-----------------------------------------------------------------------------|

|              |                                             |
|--------------|---------------------------------------------|
| Bind address | Enter the IP address of the virtual domain. |
|--------------|---------------------------------------------|

|                   |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| Server Identifier | Enter a short name for this virtual host. This is used as the directory name for web configuration files. |
|-------------------|-----------------------------------------------------------------------------------------------------------|





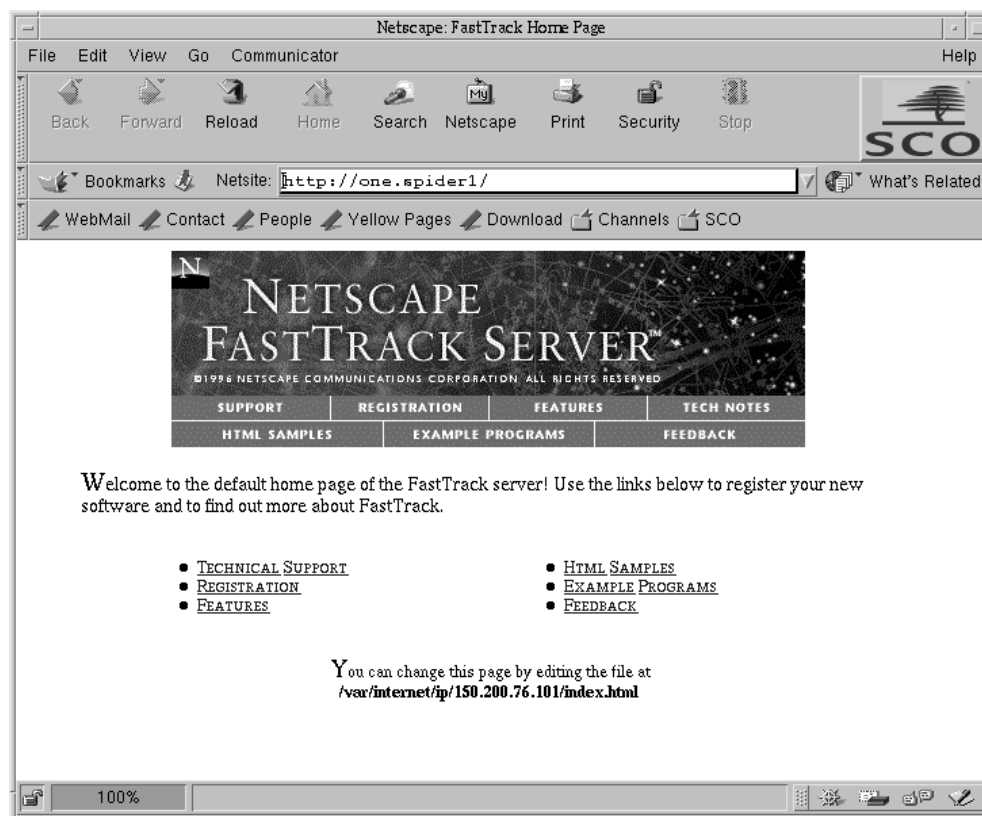
Document Root      Enter a directory to store web documents for this virtual domain. Use the default directory to store documents in `/usr/ns-home/docs`, or enter another directory. If the default is selected, all servers will share the same set of files, and serve the same home page, `/usr/ns-home/docs/index.html`.

In the example in AV 9-10 on page 9-26, documents for `web.somedom.com` are stored separately with other files used by this virtual domain in `/var/internet/ip/150.200.76.30/httpd/docs`. If this directory exists, an `index.html` file will be added automatically.

- Step 5.      Click on **OK**.
- Step 6.      Click on Start you new server when the **Success!** screen is displayed.
- Step 7.      Click on Return to the server selector. The new web server is listed using the server identifier you specified during installation.
- Step 8.      Click on the link for this new server to configure the new server. The **System Settings** page for this new server is displayed. Configuration of the web server for a virtual domain is no different than a physical host.
- Step 9.      Select **File** ⇒ **Exit** when finished.

AV 9-11

## Default Home Page



GNAD0911

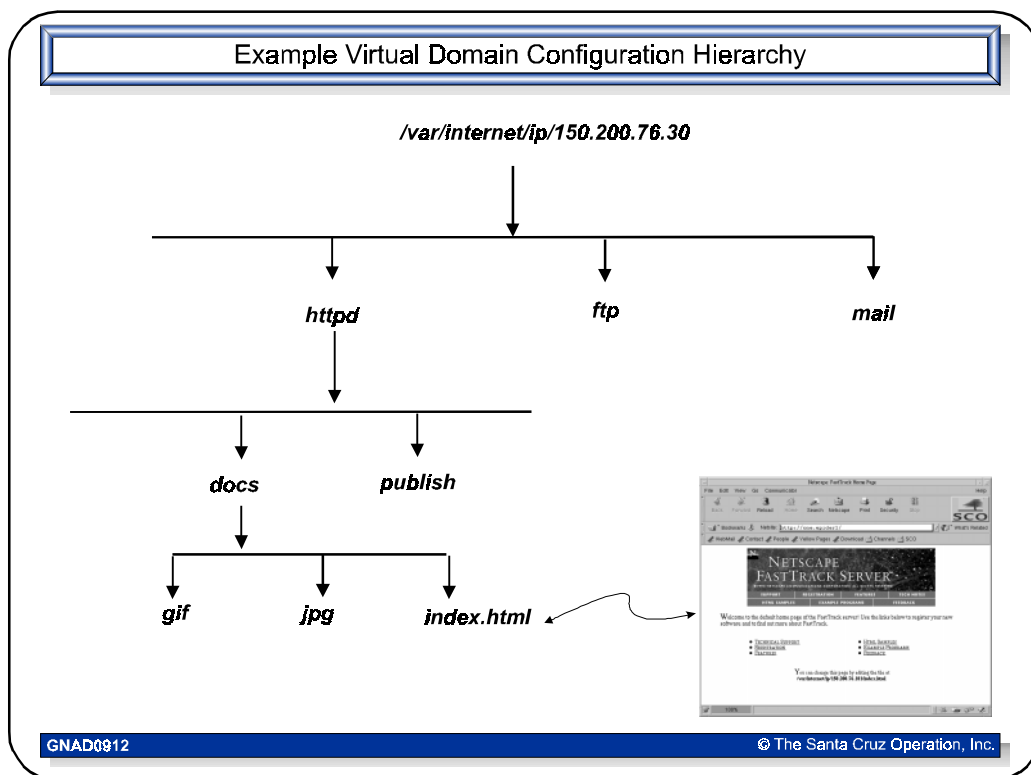
© The Santa Cruz Operation, Inc.

### **Exercise: Running the Netscape Administration Server**

---

- Step 1. Create a directory to be used as the document root location for this web server. See AV 9-12 on page 9-32 for an example.
- Step 2. Start the Netscape **Administration Server** and authenticate with user name **admin** and the original password for the **root** account.
- Step 3. Add a FastTrack server for a virtual domain. Specify the new directory as the document root location.
- Step 4. Start the server.
- Step 5. Return to the server selector.
- Step 6. Click the server link.
- Step 7. Click on Access 'server' as a Client. This opens a browser window and displays the root home page for the selected server. This is shown in AV 9-11.
- Step 8. Click on Server Off. When you see the **Success!** pop-up window, click on **OK** to confirm that the server has been shut down.
- Step 9. Access the same URL (using the **Reload** button). You can no longer access the URL.
- Step 10. Click Server On to restart the server in the **Administration Server** window.
- Step 11. Reload the URL again to verify that the server is running.
- Step 12. Return to the **Server Selector** using the URL <http://hostname:620>.

AV 9-12



### 9.3.2 Modifying the virtual domain home page

---

The default home page for new virtual domains is shown in AV 9-11. To make changes to this page, you must edit or replace the default *index.html* file for the specified virtual domain.

By default, each new instance of a web server uses */usr/ns-home/docs* as the root directory. If you did not change this directory when creating the new web server, it will serve the file */usr/ns-home/docs/index.html* by default when a client access the virtual domain web server.

Because it is likely you will want to keep separate documents for each virtual domain, create a directory hierarchy for each virtual domain and specify the directory when adding the new web server. Then, when the new server is created, a default *index.html* is added to the root document directory you specified when adding the new server.

To change the home page for a server, edit the appropriate *index.html* file. This can be done with any text editor, or using a Netscape browser in compose mode.

Use Netscape Communicator to edit the home page:

Step 1.       Select **File** ⇒ **Open Page**.

Step 2.       Enter the full path name of the *index.html* file.

Using the example in this lesson, the path is  
*/var/internet/ip/150.200.76.30/httpd/docs/index.html*.

Step 3.       Click on **Open in Composer**.

The *index.html* file is displayed and can be edited.



### **Exercise: Modifying a home page**

---

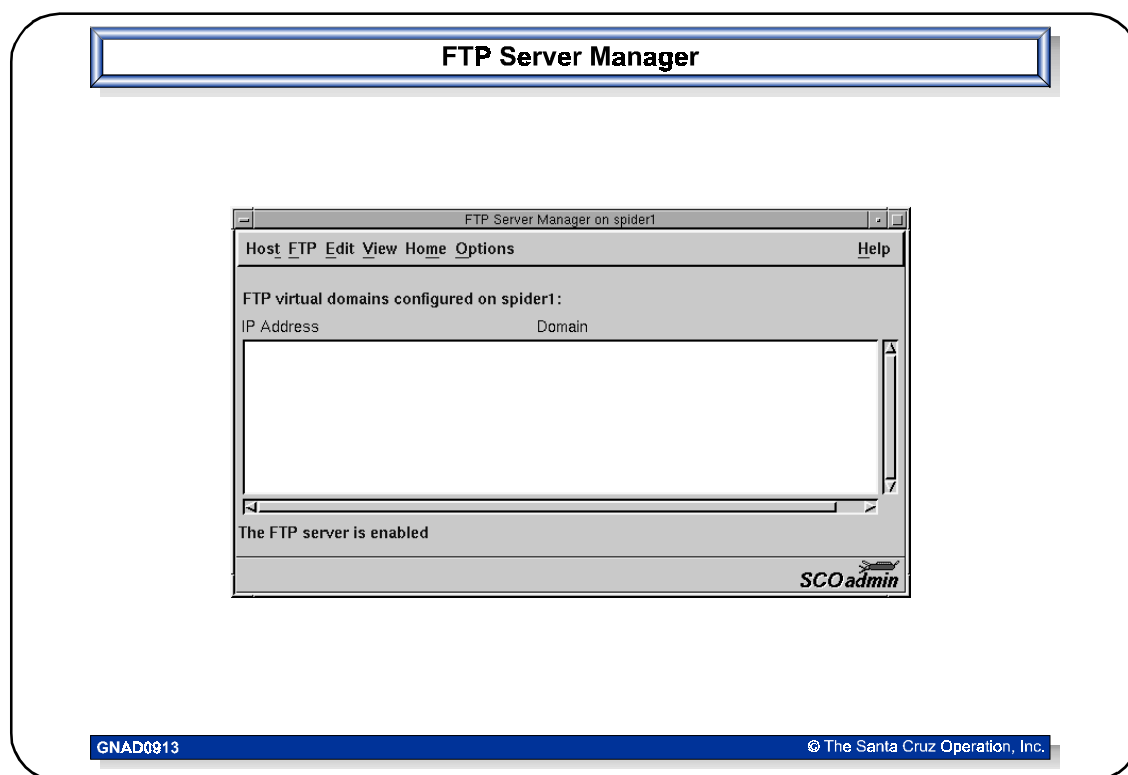
Edit the home page for a virtual domain created in previous sections. Start a Netscape browser and view the new home page by loading the URL:

*http://server*

where *server* is the host name for your virtual domain.

Note that you cannot use the host name as the URL unless DNS has been updated with the names and IP addresses of the virtual domains.

AV 9-13



GNAD0913

© The Santa Cruz Operation, Inc.



## 9.4 Configuring an FTP Server

---

The File Transfer Protocol (FTP) allows a server to be configured for incoming and outgoing file transfers. People transferring files to and from the server might have user accounts and be allowed greater access to files, or they might be unknown, and have limited access to the server through *anonymous* FTP.

The UnixWare 7 FTP server daemon, **ftpd**, allows you to configure multiple properties that affect the capabilities of a user's **ftp** session. This section discusses how to configure FTP to control access to the server, while granting both local and anonymous users the ability to transfer files.

The next section describes how FTP services are configured for virtual domains.

### 9.4.1 Enabling and disabling FTP

---

After it is enabled, FTP access is available to all users who have valid login accounts on your system. When a user logs in using **ftp(1tcp)**, their current working directory is their home directory. They can transfer files to and from directories for which they have appropriate permission.

To enable FTP access on your system:

- Step 1.        Run **scoadmin(1M)**.
- Step 2.        Select **Networking** ⇨ **FTP Server Manager**.
- Step 3.        Select **FTP** ⇨ **Server**.
- Step 4.        Toggle **Enable** or **Disable**. Note that FTP access is enabled by default.

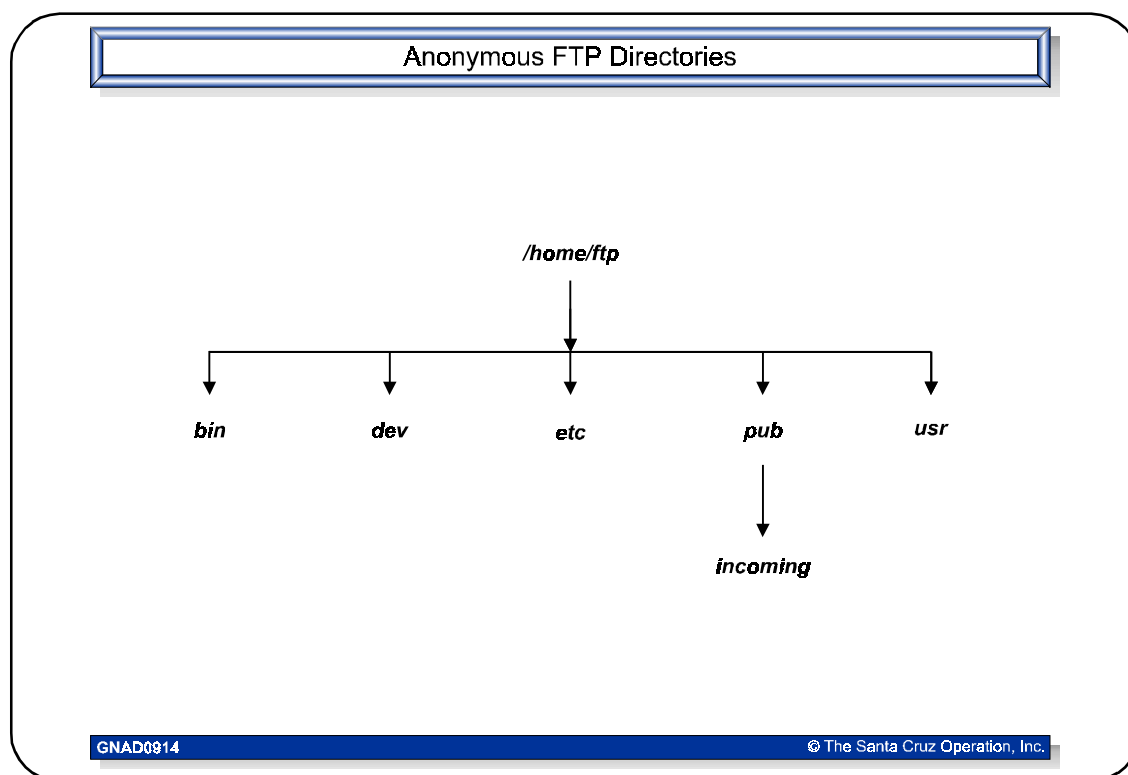
#### Exercise: Enabling FTP

---

Use the steps above to enable FTP access to your system.

Use **ftp(1tcp)** to connect to your server using an existing user account (such as **root**).

AV 9-14



## 9.4.2 Configuring anonymous FTP

---

Anonymous FTP is a common service provided by organizations with Internet connections to allow free access to a limited area of the server's filesystem. It is also used within companies to disseminate and transfer company information.

To configure anonymous FTP:

- Step 1. Start the **FTP Server Manager**.
- Step 2. Select **FTP** ⇒ **Anonymous** ⇒ **Configure**. This displays the window shown in AV 9-14.
- Step 3. Enter a directory name if the default is not acceptable.
- Step 4. Enter a user ID if the default is not acceptable.
- Step 5. Toggle “Allow incoming anonymous FTP file transfers” if you want to allow others to upload files to the incoming FTP directory.  
  
Consider that it is possible for anonymous users to fill up all available disk space if incoming transfers are allowed.
- Step 6. Click on **OK**.
- Step 7. When finished, the **FTP Server Manager** creates a file structure for FTP in */home/ftp*. See AV 9-14.

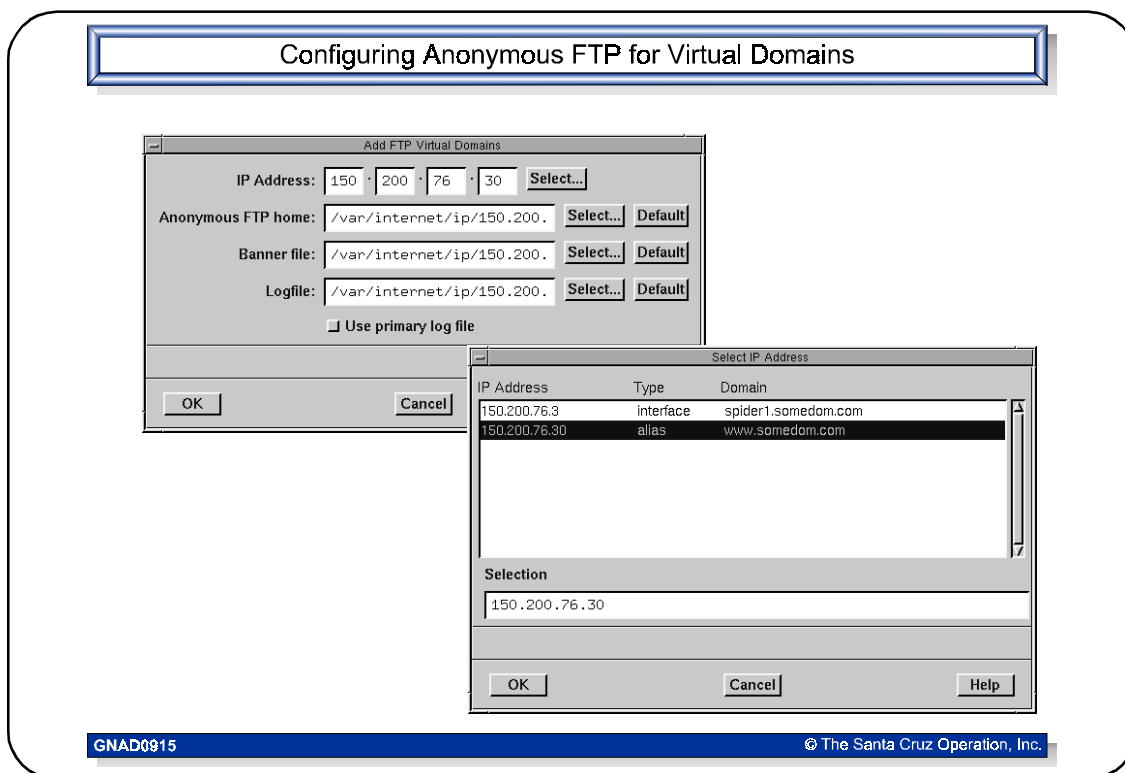


**Exercise: Configuring anonymous FTP**

---

- Step 1. Configure your server to allow anonymous users to download files using the steps in Section 9.4.2 on page 9-39.
- Step 2. View the files and directories that are created in */home/ftp*.
- Step 3. Create a dummy file in the *pub* subdirectory.
- Step 4. Use another system to connect to this server using **ftp**(1tcp). Log in as user **ftp** and use an e-mail address as your password (for example, **joe@aol.com**).
- Step 5. Use **pwd** and **ls** to see the files and directories on the server.
- Step 6. Download the dummy file. If you configured the server for writable transfers, try to upload a file to the *incoming* directory.

## AV 9-15



## 9.5 Anonymous FTP for Virtual Domains

---

Configuring anonymous FTP for a virtual domain requires two procedures:

- Enabling FTP for a virtual domain
- Setting up the FTP home directory with the files and directories that control access by anonymous users

### 9.5.1 Configuring anonymous FTP for a virtual domain

---

To enable anonymous FTP access to a virtual domain:

Step 1. Start the **FTP Server Manager**.

Step 2. Select **View** ⇒ **FTP virtual domains**.

Step 3. Select **Edit** ⇒ **Add**. Enter this information:

|            |                                                                                                     |
|------------|-----------------------------------------------------------------------------------------------------|
| IP Address | The IP address of the virtual domain. Click on <b>Select</b> to choose from an existing IP address. |
|------------|-----------------------------------------------------------------------------------------------------|

Anonymous FTP home

The path name of the root directory for anonymous FTP. Click on **Select** to choose another directory. Click on **Default** to reset to the default directory `/var/internet/ip/IP_addr/ftp/home`.

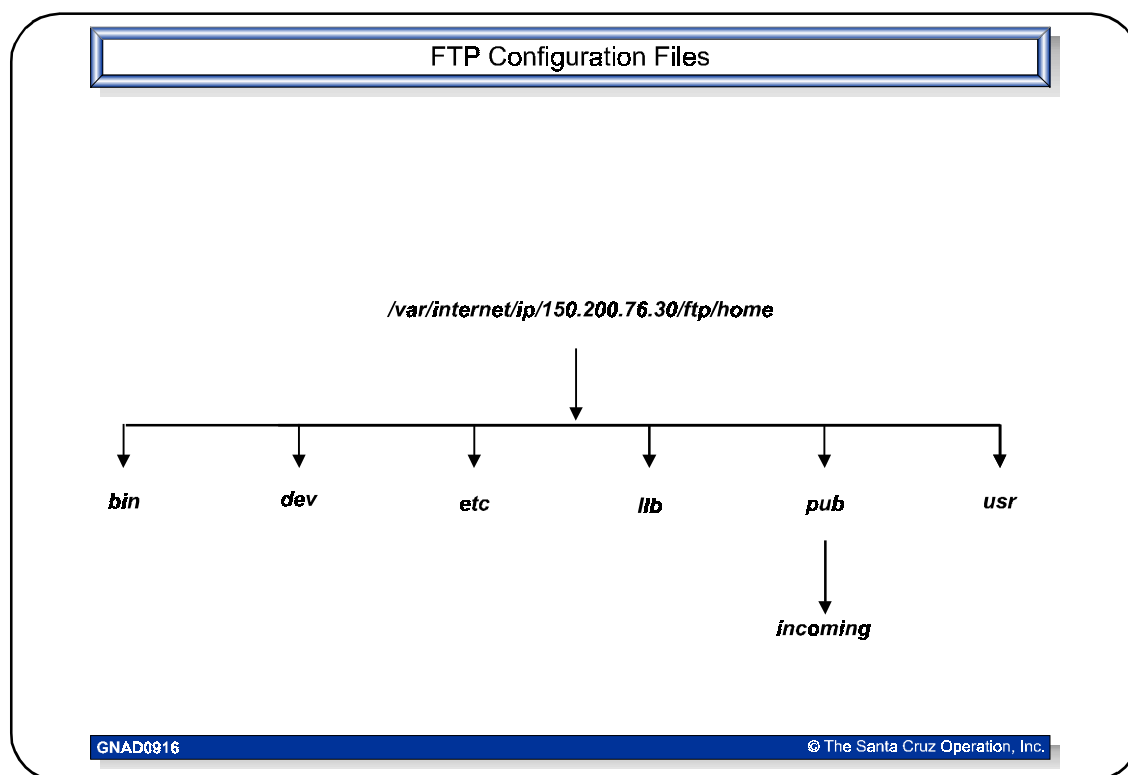
|             |                                                                                                                         |
|-------------|-------------------------------------------------------------------------------------------------------------------------|
| Banner file | The path name of the file containing the message to be displayed to users before entering their user name and password. |
|-------------|-------------------------------------------------------------------------------------------------------------------------|

|          |                                                           |
|----------|-----------------------------------------------------------|
| Log file | The path name of the file used to log FTP file transfers. |
|----------|-----------------------------------------------------------|

|                      |                                                                           |
|----------------------|---------------------------------------------------------------------------|
| Use primary log file | Toggle this to log FTP transfers to the single log for all FTP transfers. |
|----------------------|---------------------------------------------------------------------------|

Step 4. Click on **OK**.

AV 9-16





## 9.5.2 Configuring anonymous FTP directories

---

To configure anonymous FTP directories for a virtual domain:

- Step 1. Start the **FTP Server Manager**.
- Step 2. Select **View** ⇒ **FTP virtual domains**.
- Step 3. Select the IP address of the virtual domain.
- Step 4. Select **Home** ⇒ **FTP Virtual Domains** ⇒ **Install**. Enter this information:
- |                          |                                                                                    |
|--------------------------|------------------------------------------------------------------------------------|
| Domain names             | Toggle to select a virtual domain by name                                          |
| IP addresses             | Toggle to select a virtual domain by name                                          |
| Allow incoming transfers | Toggle “off” unless you want to allow anonymous users to write files to the server |
- Step 5. Click on **OK**.
- Step 6. Click on **OK** when the **Home Directory Report** screen is displayed

### **Exercise: Configuring anonymous FTP for a virtual domain**

---

- Step 1. Enable anonymous FTP for the virtual domain you created.
- Step 2. Configure the anonymous FTP home directory.
- Step 3. View the directories that have been created in */var/internet/ip/IP\_addr/ftp/home*.
- Step 4. Create a dummy file in the *pub* subdirectory.
- Step 5. Use **ftp**(1M) to connect to a virtual domain. Enter **anonymous** or **ftp** at the **Name** prompt and an e-mail address as your password (for example, *joe@aol.com*).
- Step 6. Use **pwd** and **ls** to see the files and directories on the server.
- Step 7. Download the dummy file. If you configured the server for writable transfers, try to upload a file to the *incoming* directory.

## 9.6 Other FTP configurations

---

You can configure FTP to deny, allow, or limit access based on a number of different criteria. The next sections discuss configuration options that determine the type of access a user is granted, and the tools available for transferring files.

### 9.6.1 Limiting access using classes

---

An FTP user class defines a set of users according to the account type they are accessing and the IP address of the machine from which they are connecting.

Account types include:

|                  |                                                                                                                                                                                                                                |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Real</i>      | A user is considered <i>real</i> when the account name and password provided at authentication time are those that exist in the <i>/etc/passwd</i> and <i>/etc/shadow</i> file, and the user is not a member of a guest group. |
| <i>Guest</i>     | A user is considered a <i>guest</i> when the account name provided at authentication time is a member of a guest group.                                                                                                        |
| <i>Anonymous</i> | A user is considered <i>anonymous</i> when the account name provided at authentication time is either <b>ftp</b> or <b>anonymous</b> .                                                                                         |

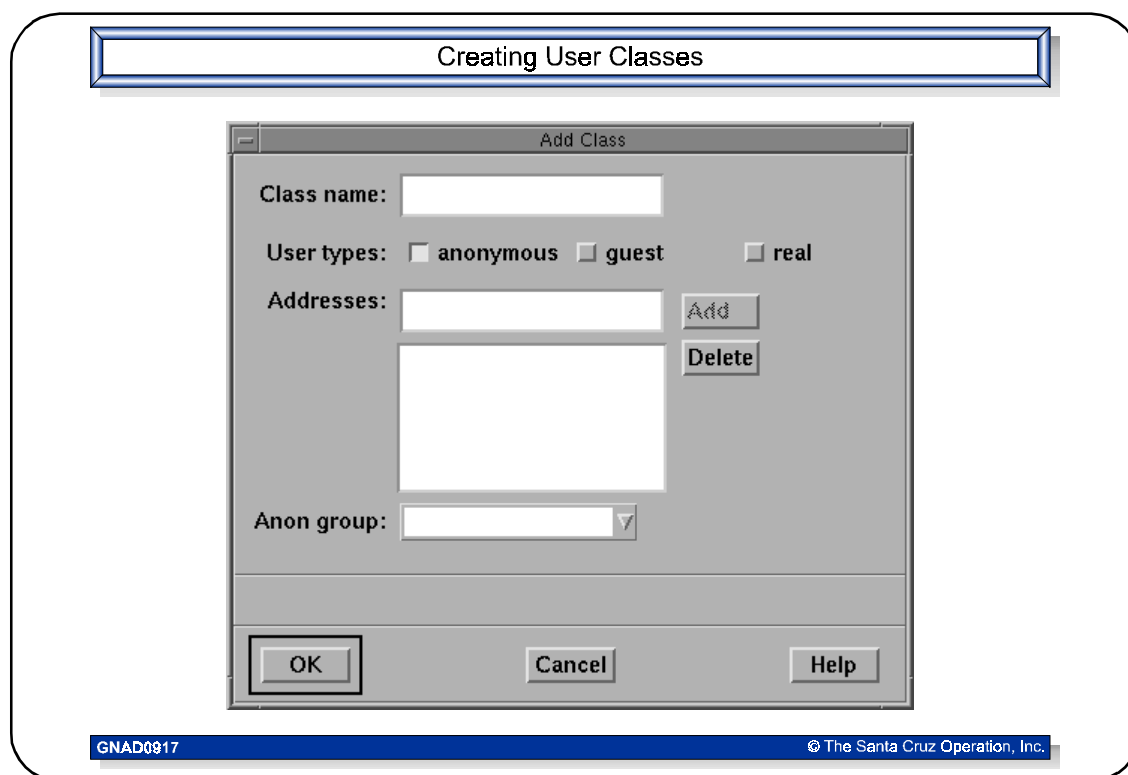
A user class also allows you to control how many sessions you are willing to accept from users belonging to each class. These limits are called *class limits*.

Every user that tries to access your host using FTP must belong to a user class. At ISL, a class called **all** exists that includes anybody trying to access your host through FTP. You might want to limit how people can access your host using FTP. Classes are used to create these limits.

Create user classes by either:

- Using **vi**(1) to edit the *ftppaccess*(4) file according to the manual page
- Using the **FTP Server Manager**

AV 9-17



To create user classes using the **FTP Server Manager**:

- Step 1. Run **scoadmin(1M)**.
- Step 2. Select **FTP Server Manager** ⇒ **View** ⇒ **Classes**.
- Step 3. Select **Edit** ⇒ **Add** to display the window in AV 9-17.
- Step 4. Complete the form as appropriate.
- Step 5. Click on **OK** to create the user class.

### **Exercise: Creating a user class**

---

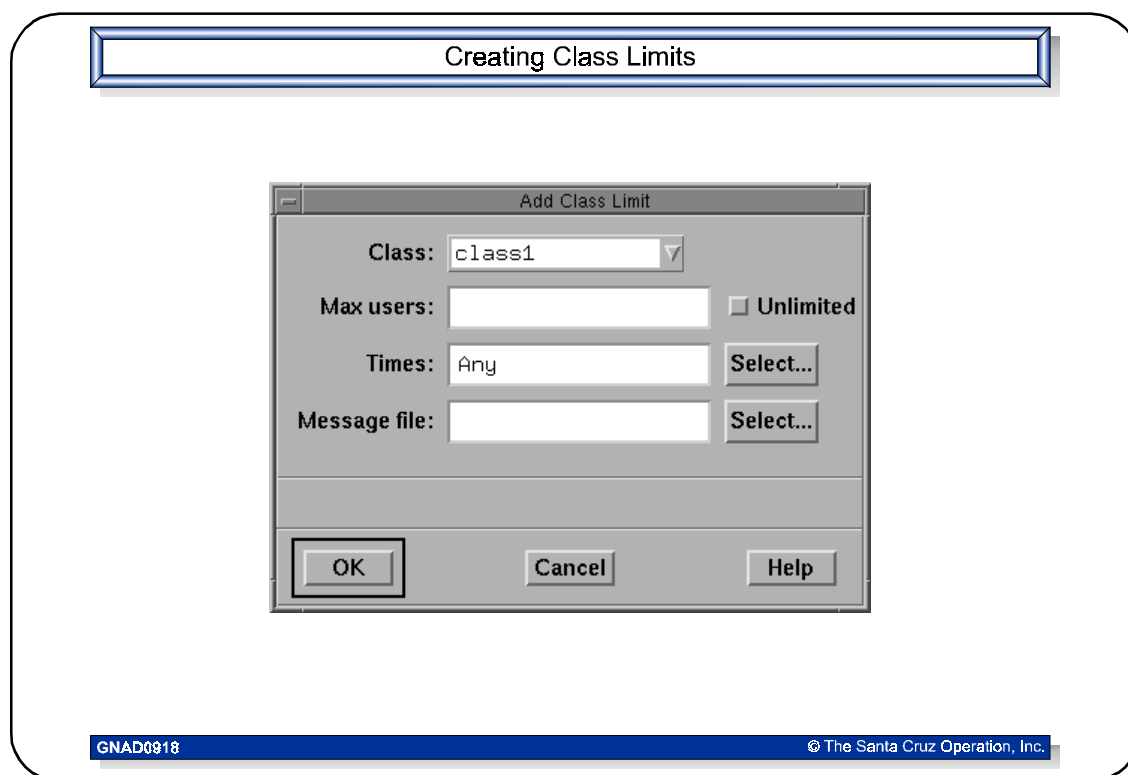
Using the **FTP Manager**, create a class called *class1* that only applies to:

- Anonymous and guest users
- Users coming from your host's domain

- Step 1. Run **scoadmin(1M)**.
- Step 2. Select **FTP Server Manager** ⇒ **View** ⇒ **Classes**.
- Step 3. Select **Edit** ⇒ **Add**.
- Step 4. Enter **class1** in the "Class name" field.
- Step 5. Click on the **anonymous** and **guest** buttons.
- Step 6. Enter your host's domain name into the "Addresses" field.
- Step 7. Click on **Add**.
- Step 8. Click on **OK** to create the user class.

For more information, refer to **scohelp** ⇒ **Intranet and Internet** ⇒ **Configuring File Transfer Protocol (FTP) servers** ⇒ **Configuring User Classes**.

AV 9-18



Create class limits by either using:

- **vi(1)** to edit the *ftpaccess(4)* file according to the manual page
- The **FTP Server Manager**

To create user class limits using the **FTP Server Manager**:

- Step 1.        Run **scoadmin(1M)**.
- Step 2.        Select **FTP Server Manager** ⇒ **View** ⇒ **Class Limits**.
- Step 3.        Select **Edit** ⇒ **Add**. This displays the window in AV 9-18
- Step 4.        Complete the form as appropriate.
- Step 5.        Click on **OK** to create the user class limit.

For more information, refer to **scohelp** ⇒ **Intranet and Internet** ⇒ **Configuring File Transfer Protocol (FTP) servers** ⇒ **Setting class limits**.

### **Exercise: Adding a Class Limit**

---

Use the **FTP Manager** to create a class limit that:

- Applies to the class *class1*.
- Limits the number of users to two (2).

- Step 1. Run **scoadmin(1M)**.
- Step 2. Select **FTP Server Manager** ⇨ **View** ⇨ **Class Limits**.
- Step 3. Select **Edit** ⇨ **Add**.
- Step 4. Select **class1** from the “Class” menu.
- Step 5. Enter **2** into the “Max users:” field.
- Step 6. Click on **OK** to create the class limit.
- Step 7. Test the configuration by attempting three (3) simultaneous FTP sessions to your machine as the **anonymous** user.



## 9.6.2 Other FTP configurations

---

An alternative to using the **FTP Manager** is to use the command line and configuration files. Most of the files were discussed in previous sections of the course.

The following sections describe:

- Files that have not yet been discussed that are not configurable using the **FTP Server Manager**.
- Elements in files where the files have been discussed but the elements have not.

### 9.6.2.1 Denying access to FTP by username or shell

---

You may control what user names might be used to access your FTP server. This is commonly used to prevent users from accessing your FTP server as **root** and other sensitive accounts.

The manual page for **ftpd**(1Mtcp) indicates that account names presented must:

- Exist on the system, or be either **anonymous** or **ftp** if Anonymous FTP is configured
- Not have a null password
- Not exist in the */etc/ftpusers* file, unless it is **anonymous** or **ftp**
- Have a login shell that exists in the */etc/shells* file

The */etc/shells* file:

- Has one entry per line
- Has the name of a different shell (e.g. */bin/ksh*) on each line

#### **Exercise: Viewing the */etc/shells* file**

---

Use the **cat**(1) command to view the default entries in the */etc/shells* file.

### 9.6.2.2 FTP conversions and the *ftpconversions* file

---

To increase the efficiency of file transfers, some files or directories might need to be converted before they are transferred. This is done to save bandwidth and reduce transmission times.

Conversions are configured in the */etc/ftpconversions* file which is documented in the *ftpconversions(4tcp)* manual page. Modify this file if you need to add conversions not present by default. The conversions present by default include:

- Compression with **compress**(1), **gzip**, and **zip**
- Archiving directories using **tar**(1) and **cpio**(1)
- Combining any of the above

For example, suppose that there is a file called *foo* and that the default conversions are in place. A person using a FTP client enters:

**get foo.Z**

This causes the FTP daemon to compress (using **compress**(1)) and send the *foo* file. Bandwidth is saved and the transfer time reduced because a smaller version of the file is being transferred. It is the destination user's responsibility to uncompress the file.

As another example, suppose that there is a directory called *bar* and that the default conversions are in place. A person using an FTP client enters:

**get bar.tar.Z**

This causes the FTP daemon to send a compressed **tar**(1) archive of the *bar* directory. In addition to the bandwidth savings, this conversion also makes it easier for the user to get an entire directory with one command. It is the destination user's responsibility to uncompress and extract the **tar** file.

## AV 9-19

Default `/etc/ftppass` File

```
#ident "(#)ftppass 1.3"
#
ftpd configuration file, see ftpaccess(4tcp).
#
loginfails 3
passwd-check trivial warn
class all anonymous,guest,real *
compress yes all
tar yes all
readme README* login all
readme README* cwd=* all

#banner /var/ftp/banner.msg
#message /var/ftp/welcome.msg login all
#message .message cwd=* all

#chmod no anonymous
#delete no anonymous,guest
#overwrite no anonymous,guest
#rename no anonymous,guest
#umask no anonymous

#limit all 5 Wk0900-1800 /var/ftp/toomany.msg
#limit all 20 SaSu|Any1800-0900 /var/ftp/toomany.msg

#log commands anonymous,guest,real
#log transfers anonymous,guest,real inbound,outbound

#path-filter anonymous,guest /var/ftp/filename.msg ^[:alnum:]-._]*$ ^[-.]

#upload /home/ftp * no nodirs
#upload /home/ftp /pub/incoming yes ftp other 0444 nodirs
```

GNAD0919

© The Santa Cruz Operation, Inc.

### 9.6.2.3 The *ftppaccess* file

---

Most FTP configuration is done through the **FTP Server Manager**, which result in entries being placed in the */etc/ftppaccess* file.

You only need to modify this file manually if you cannot make the necessary modifications using the **FTP Server Manager**.

AV 9-19 shows the *ftppaccess* file that is installed by default. It contains these entries:

|                           |                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>loginfails</code>   | Allows users to log into your ftp server three consecutive times unsuccessfully before they are forced to disconnect and try again.                                                                                                                                                                                                                             |
| <code>class</code>        | Defines a class of users called <i>all</i> that includes every possible user type (real, guest, and anonymous) logging in from anywhere (*). A reasonable change for security purposes is to declare two classes of users, those inside your domain, and those outside your domain. The latter class can then be carefully limited in their access and numbers. |
| <code>passwd-check</code> | Defines the level and enforcement of password checking for anonymous FTP.                                                                                                                                                                                                                                                                                       |
| <code>compress</code>     | Transparently compresses files before transfer. See Section 9.6.2.2, “FTP conversions and the <i>ftpconversions</i> file,” on page 9-55 for details.<br><br>This can significantly reduce Internet traffic and download times.                                                                                                                                  |
| <code>tar</code>          | Allows a user to get an entire directory from the server with one command. See Section 9.6.2.2, “FTP conversions and the <i>ftpconversions</i> file,” on page 9-55.                                                                                                                                                                                             |

**readme**

If there is a file beginning with *README* in a directory, a message indicating the last time it was changed, and requesting the user read this file, is displayed at login time or upon using the **cd** command.

The first **readme** entry in AV 9-19 causes the *README* in the **ftp** users's home directory to have a message displayed at login time. The second entry causes any given directory's *README* to be displayed the first time a user performs a **cd** to that directory. You would typically put a description of directory contents into each *README*.

If the asterisk (\*) is included, all files beginning with *README* are displayed. Also, path names can be included in **readme** entries. For anonymous users, the path must be relative to the base of the anonymous FTP directory tree.

There are several other entries that can be specified in this file. Those most useful for security include: **chmod**, **delete**, **overwrite**, **log commands**, and **log transfers**. Another useful entry is **banner**, but check the manual page before using it

.

---

**Exercise: Modifying *ftppaccess(4tcp)***

---

Modify the */etc/ftppaccess* file on your system to:

- Display *.message* files in any directory upon login
- Limit the total number of FTP clients to one, displaying an appropriate message if this limit is exceeded

Test your configuration by attempting two simultaneous FTP sessions to your server.

### 9.6.2.4 The */etc/ftpgroups* file

---

The UnixWare 7 FTP server supports an extension that allows anonymous users to change group membership during their FTP session. Configuring this extension allows you to create password-protected areas of your anonymous FTP files.

To enable this extension:

- Step 1. Set the `private` option in the `ftpassess(4tcp)` file to `/etc/ftpgroups`, if it is not already set.
- Step 2. Create the `/etc/ftpgroups` file and add an entry that maps a pseudo-group to a real group. For example, the entry:

```
privgroup:abcd1234ask32:sys
```

- Maps the pseudo-group *privgroup* to the real group *sys*
- Gives the pseudo-group a password that encrypts to `abcd1234ask32`

Use **vi**(1) to create the entry in the `/etc/ftpgroups` file, minus the password field. Note that you might have to use **chmod**(1) to add write permission.

To obtain an encrypted password string that is added to `/etc/ftpgroups`, use the **passwd**(1) command to set the password on some user account (any account—use **root**, or create a new dummy account if desired). Then use **vi** to copy-and-paste the password field from the `/etc/shadow` file to the `/etc/ftpgroups` file.

For example, to copy **root**'s password:

- Enter `vi /etc/shadow /etc/ftpgroups`.
- Place the cursor on the first character of **root**'s encrypted password.
- Yank **root**'s password into the buffer by typing **yt :** without pressing .
- Switch to the other file by entering **:n** .
- Position the cursor at the first colon in the entry.
- Paste the contents of buffer a by typing **p**.

- Step 3. Change the group ownership to the **sys** group for files you only want members of the **privgroup** pseudo-group to access

To use this feature from a FTP client:

- Step 1. Start **ftp**.

- Step 2. Authenticate as user **ftp**.

- Step 3. Enter this command:

**site group privgroup**

- Step 4. Enter this command:

**site gpass *password***

where *password* is the password of the user copied from the */etc/shadow* file in Step 2 on the previous page.

- Step 5. For the rest of the FTP session, the user can access files owned by the group **sys**.



### 9.6.3 Flags for **ftpd(1Mtcp)**

---

There are many flags for **ftpd** that can be of use to you as a network administrator. Logging sessions and writing debugging information are just two types of flags that can be set. Refer again to the manual page for **ftpd(1Mtcp)** to review each of these flags with your instructor.

#### **Exercise: Modifying the existing ftpd entry**

---

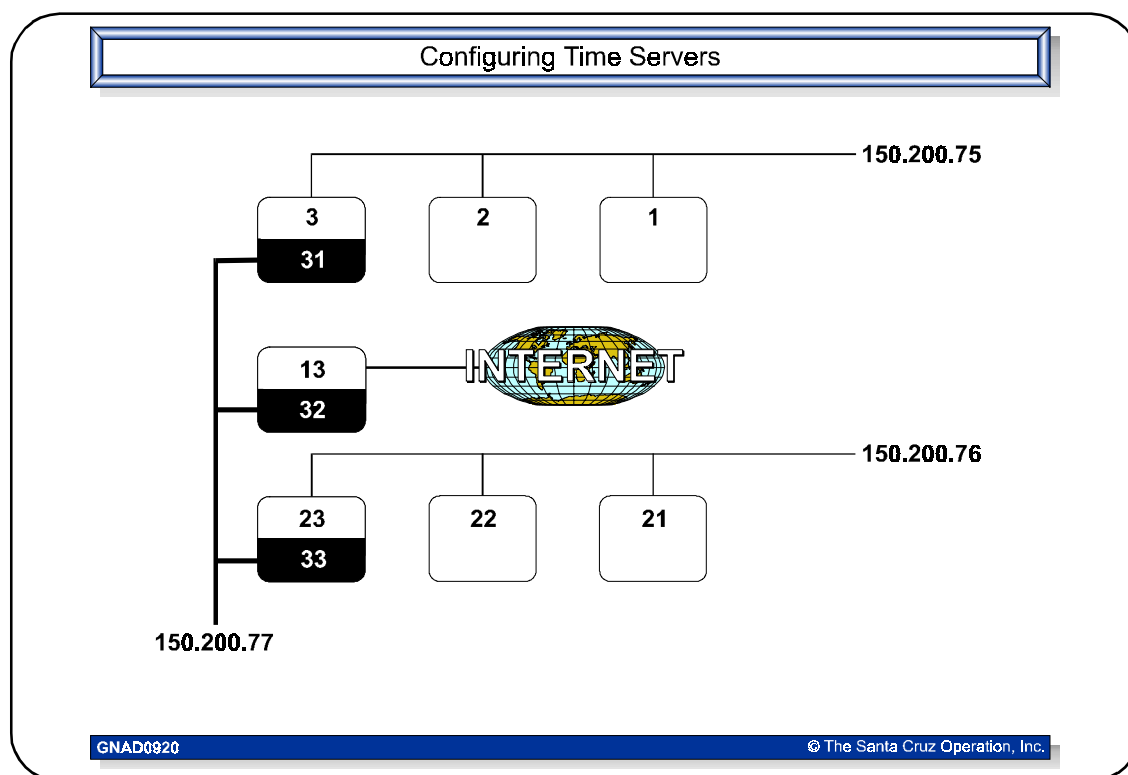
Use the manual page to determine which command-line flags cause **ftpd(1Mtcp)** to log all commands received from remote FTP clients.

Modify the `/etc/inet/inetd.conf` file to cause **ftpd** to log all FTP sessions to the `/var/adm/log/osmlog` file.

Stop and restart the **inetd** service.

Verify that **ftpd** is now logging.

AV 9-20



## 9.7 Configuring Time Synchronization

---

Whenever hosts in a network are performing client/server operations that exchange time-oriented data, all of the hosts must agree on the current date and time.

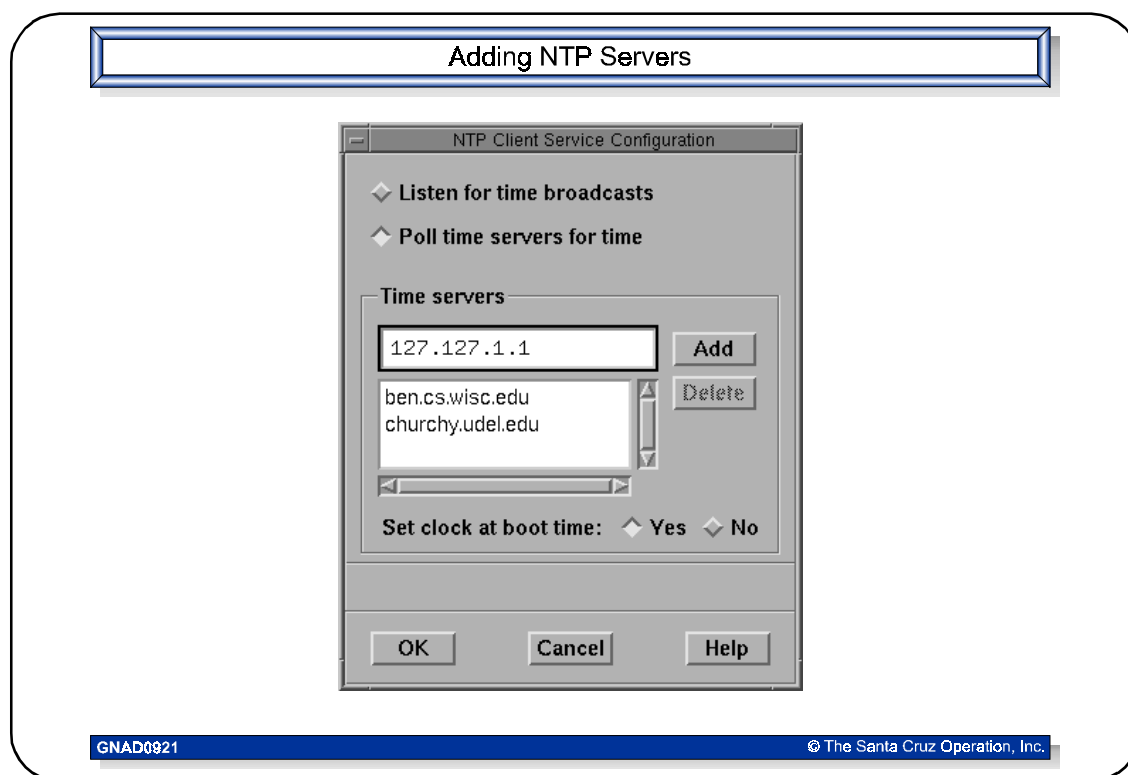
The **xntpd**(1Mtcp) daemon provides an easy way to have your hosts agree on the current date and time. The **xntpd** daemon implements the Network Time Protocol (NTP). In NTP:

- There are hosts on the Internet, throughout the world, that are special, highly reliable *time servers*. Some of these servers are synchronized with atomic clocks.
- The farther away (in network hops) a time server is from an atomic clock, the less reliable it is. This reliability is called the server's *stratum*.
- Stratum 1 servers are called primary servers. Stratum 2 servers are called secondary servers. Local clocks default to stratum 3, but can and should be set to stratum 10.
- For every 100 hosts on your intranet, you should configure one NTP client to poll a strata 1 server and a stratum 2 server (in the event the stratum 1 server is down).
- You should configure all other hosts in your intranet to poll your local time servers.
- You should configure all machines to poll their local clock in case all other sources of time fail.
- If you configure a host as a NTP client running **xntpd**, it will act as an NTP server as well.

For example, for the intranet shown in AV 9-20, configure:

- Host 13/32 to poll a primary and secondary server
- All other hosts to poll host 13/32

AV 9-21



The configuring hosts to run **xntpd**(1Mtcp):

- Step 1. Determine which hosts on your intranet will poll primary and secondary servers.
- Step 2. Determine which primary and secondary servers to poll from <http://www.eecis.udel.edu/~mills/ntp/servers.html>.
- Step 3. Add appropriate entries to the */etc/inet/ntp.conf* file on each host using the **Client Manager**. In addition to remote time servers, add the local clock at address 127.127.1.1.
- Step 4. Edit the */etc/inet/ntp.conf* file, adding a line that sets the stratum of the local clock to 10. Do this by adding the following line after the line listing the 127.127.1.1 server:  
  
**fudge 127.127.1.1 stratum 10**
- Step 5. Signal **xntpd** to re-read the *ntp.conf* file.

**Exercise: Configuring a host to poll an NTP server**

---

- Step 1. Start the **Client Manager**.
- Step 2. Select **Services** ⇒ **Add** ⇒ **NTP client**.
- Step 3. Enter the name of the instructor's machine in the text field next to the **Add** button.
- Step 4. Click on **Add**.
- Step 5. Repeat steps 6 and 7 using the local clock at address 127.127.1.1.
- Step 6. Click on **OK**.
- Step 7. Select **Host** ⇒ **Exit** to exit the **Client Manager**, update the */etc/ntp.conf* file, and start **xntpd**.
- Step 8. Edit the */etc/inet/ntp.conf* file, adding a line that sets the stratum of the local clock to 10. Do this by adding the following line after the line listing the 127.127.1.1 server:
- fudge 127.127.1.1 stratum 10**
- Step 9. Signal **xntpd** to re-read the *ntp.conf* file:
- Find the **xntpd** process ID:  

```
ps -ef | grep xntpd
```
  - Signal the process:  

```
kill -1 PID
```
- Step 10. Use the **ntpq** and **xntpd** commands to list the server this host is currently polling.
- Step 11. Monitor the time on your host and the instructor's to verify that they converge.

## 9.8 Lesson Summary

---

---

This lesson has discussed the configuration of Internet services, focusing on web and FTP.

### Review questions

---

1. Do all TCP/IP-based daemons support virtual domains?
2. Which command associates an IP alias with a network adapter?
3. What is HTTP and how does it relate to the World Wide Web?
4. In which file does the **FTP Server Manager** record most of its configuration information?
5. Which file contains configuration information for the **xntpd**(1Mtcp) daemon?

## 9.9 Relevant Commands and Files

---

The following commands and files were presented in this lesson.

### Relevant commands

---

|                          |                                                                                   |
|--------------------------|-----------------------------------------------------------------------------------|
| <b>compress(1)</b>       | Compresses files. Used in <i>ftpconversions(4tcp)</i>                             |
| <b>cpio(1)</b>           | Is the copy-in-out archiving command. Used in <i>ftpconversions(4tcp)</i>         |
| <b>ifconfig(1Mtcp)</b>   | Configures network interfaces and IP aliases                                      |
| <b>in.rlogind(1Mtcp)</b> | Is the server daemon for remote logins                                            |
| <b>in.ftpd(1Mtcp)</b>    | Is the server daemon for FTP sessions                                             |
| <b>in.telnetd(1Mtcp)</b> | Is the server daemon for <b>telnet</b> sessions                                   |
| <b>netscape(X1)</b>      | Is the browser used to access the Netscape Administration Server                  |
| <b>nsfast(1M)</b>        | Controls Netscape FastTrack server daemons                                        |
| <b>sacadm(1M)</b>        | Administers services controlled by the Service Access Facility (SAF)              |
| <b>scoadmin(1M)</b>      | Is the interface-independent program that provides access to the various managers |
| <b>scohelp(1X)</b>       | Provides access to web-based help and manual pages                                |
| <b>sendmail(1M)</b>      | Is the server daemon for e-mail (an MTA)                                          |
| <b>tar(1)</b>            | Creates tape archives (used in <i>ftpconversions(4tcp)</i> )                      |
| <b>xntpd(1Mtcp)</b>      | Implements the NTP protocol                                                       |



## Relevant files

---

|                                |                                                                                                                                                                                           |
|--------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>/etc/ftpaccess</i>          | Contains parameters that control most of the configurable behavior of <b>in.ftpd</b> (1Mtcp). Documented in <i>ftpaccess</i> (4tcp).                                                      |
| <i>/etc/ftpconversions</i>     | Contains rules for converting (compressing and archiving) files during FTP sessions. Documented in <i>ftpconversions</i> (4tcp).                                                          |
| <i>/etc/ftphosts</i>           | Contains a list of hosts from which FTP connections are refused. Documented in <i>ftphosts</i> (4tcp).                                                                                    |
| <i>/etc/ftpgroups</i>          | Contains a list of pseudo-group to real-group mappings for use with the SITE extended FTP commands. Documented in <i>ftpgroups</i> (4tcp).                                                |
| <i>/etc/ftpusers</i>           | Contains a list of users which cannot be used to authenticate an FTP session. Documented in <i>ftpusers</i> (4tcp).                                                                       |
| <i>/etc/group</i>              | Contains a list of real groups. Documented in <i>group</i> (4).                                                                                                                           |
| <i>/etc/inet/ntp.conf</i>      | Contains the configuration information for the <b>xntpd</b> (1Mtcp) daemon                                                                                                                |
| <i>/etc/passwd</i>             | Contains a list of real users and their home directories. Documented in <i>passwd</i> (4).                                                                                                |
| <i>/etc/rc2.d/S70ipaliases</i> | Is the possible location for permanent IP aliases for virtual domains if they are not configured in the <i>config</i> (4tcp) file. Contains one or more <b>ifconfig</b> (1Mtcp) commands. |
| <i>/etc/services</i>           | Is a list of names for server daemons, their protocol, and their port number. Documented in <i>services</i> (4tcp).                                                                       |
| <i>/etc/shadow</i>             | Contains a list of real users and their encrypted passwords. Documented in <i>shadow</i> (4).                                                                                             |
| <i>/etc/shells</i>             | Contains a list of login-shells a real user must have listed in the last field of their <i>passwd</i> (4) entry in order for authentication to succeed                                    |
| <i>/home/ftp</i>               | Is the default home directory of the <b>ftp</b> user                                                                                                                                      |
| <i>/usr/ns-home</i>            | Is the default home directory for Netscape FastTrack files                                                                                                                                |

|                                   |                                                                                                                 |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <i>/usr/ns-home/cgi-bin</i>       | Is the default directory for CGI files served by the Netscape FastTrack server                                  |
| <i>/usr/ns-home/docs</i>          | Is the default directory for HTML pages served by the Netscape FastTrack server                                 |
| <i>/usr/ns-home/httpd-80</i>      | Is the default directory for files configuring the behavior of the Netscape FastTrack server running on port 80 |
| <i>/usr/ns-home/httpd-80/logs</i> | Is the default directory for log files for the Netscape FastTrack Server running on port 80                     |
| <i>/usr/ns-home/start-admin</i>   | Starts the Netscape Administration Server daemon                                                                |
| <i>/usr/ns-home/stop-admin</i>    | Stops the Netscape Administration Server daemon                                                                 |

## 9.10 Answers to Review Questions

---

---

1. Support of virtual domains is up to the individual daemons.
2. The **ifconfig**(1Mtcp) command associates an IP alias with a network adapter.
3. HTTP is the HyperText Transport Protocol and it is the language spoken by all web servers on the World Wide Web.
4. The **FTP Server Manager** records most of its configuration information in the */etc/ftpaccess* file.
5. The */etc/inet/ntp.conf* file contains the configuration information for the **xntpd**(1Mtcp) daemon.

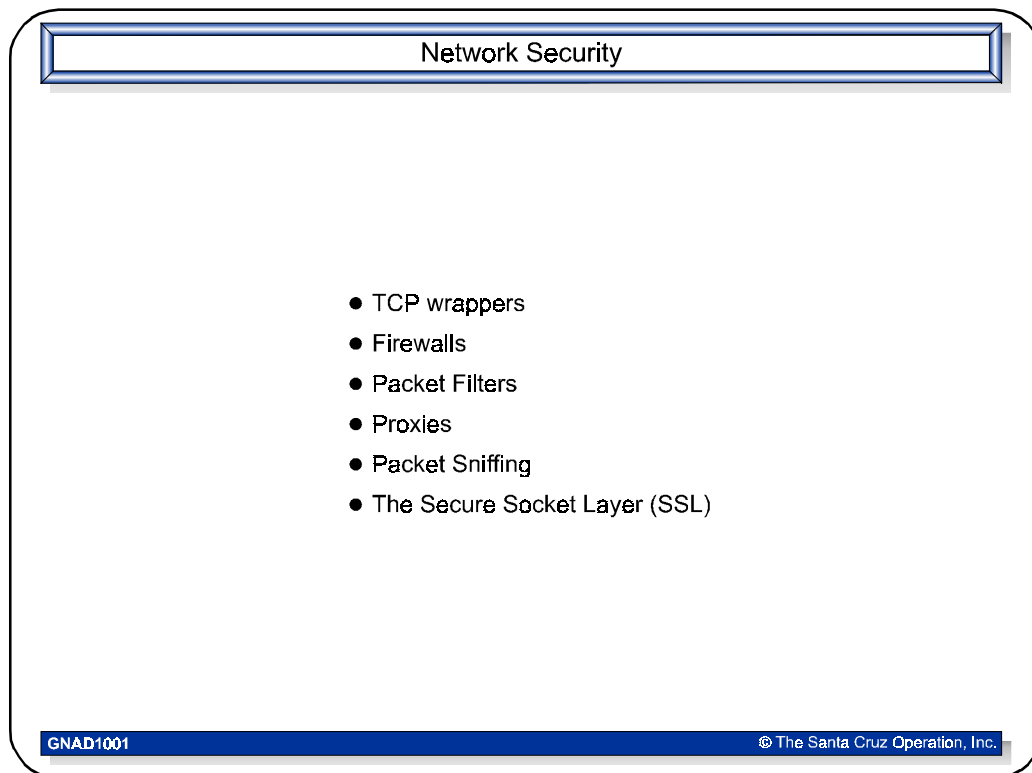


---

# ***10***

## ***Network Security***

AV 10-1



**Lesson 10**

## **Network Security**

### **Introduction**

---

We have previously discussed how to connect your machine to a local area network and to the Internet, and how to provide services to the network. These connections provide access to information on your system to a variety of users. Unfortunately, they also open your system up to attack from external and often anonymous sources.

A breach of security can cause a loss of service or data that can be both damaging and costly.

In this section, various security measures are discussed that can be used to secure your system and to thwart those who intend to damage or misuse your system.

UnixWare 7 has powerful built-in network security features such as TCP wrappers and packet filters. This lesson discusses how to configure and use these features to protect your systems. Other security precautions, such as packet sniffers and proxy servers, are also discussed.

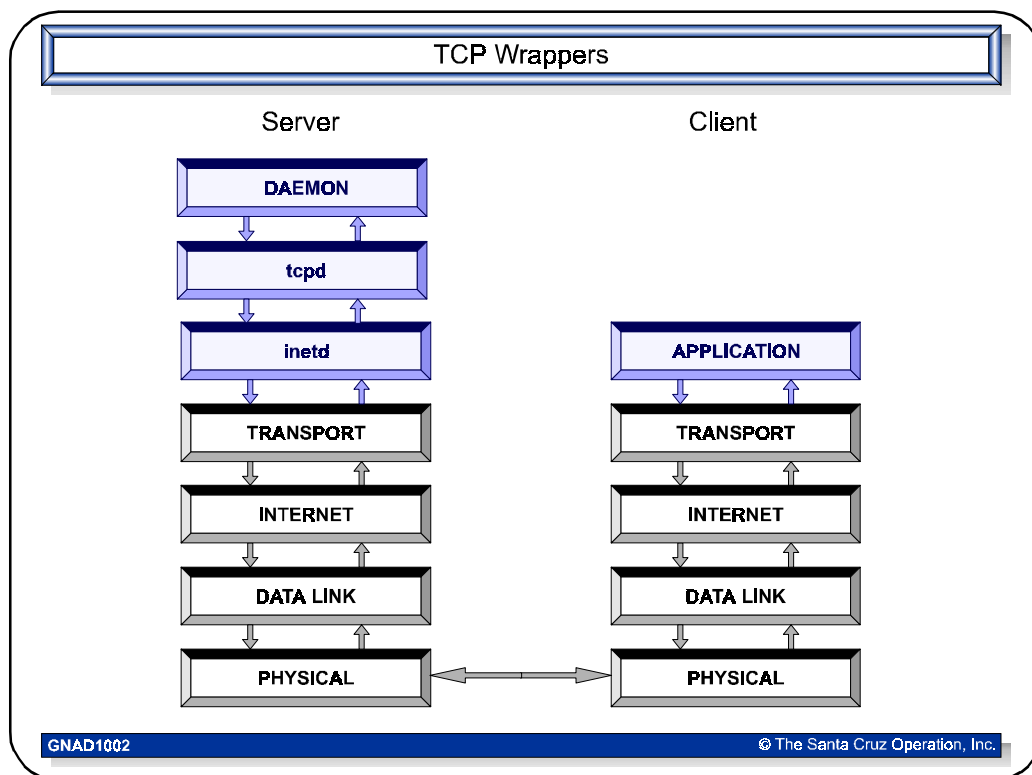
### **Objectives**

---

This lesson discusses network security features and describes how to configure those built into UnixWare 7. By the end of this lesson, you will be able to:

- Configure TCP wrappers to control access to network services on your system
- Configure packet filters to restrict the flow of packets through your system
- Discuss the use of firewalls
- Describe the function of proxy servers
- Discuss the dangers and uses of packet sniffers and how to protect against them by using SSL

## AV 10-2





## 10.1 TCP Wrappers

---

UnixWare 7 includes a facility called *TCP wrappers*. TCP wrappers allows you to control access by hosts to service daemons that are started by **inetd**(1Mtcp). These service daemons include **in.ftpd**(1Mtcp), **in.rlogind**(1Mtcp), and **in.telnetd**(1Mtcp). You can configure, on a per-service basis, which hosts or domains are allowed access to services on your system.

TCP wrappers, **inetd**, and the service daemon are implemented at the application layer of the IP stack. This is shown in AV 10-2.

The principle that TCP wrappers uses to operate is simple:

1. The server receives a request for a service from a client host.
2. Using an appropriately modified entry in the *inetd.conf*(4tcp) configuration file, **inetd** invokes **in.tcpd**(1Mtcp) the wrapper daemon, instead of starting the daemon for the service being requested.
3. The **in.tcpd** daemon consults the lists of rules in the */etc/inet/hosts.allow* and then in the */etc/inet/hosts.deny* files.
4. If the host is:
  - Allowed access to a service in the *hosts.allow* file, **in.tcpd** invokes the service daemon immediately
  - Denied access to the service in the *hosts.deny* file, **in.tcpd** does not invoke the service daemon
  - Not covered by a rule in either the *hosts.allow* or *hosts.deny* files, it is implicitly allowed, and **in.tcpd** invokes the service daemon
5. The connection request is logged by **syslogd**(1M).

By default, many service daemons are started by **in.tcpd** instead of **inetd**. You can control host access immediately to these services by adding appropriate entries to the files:

- *hosts.allow*(4tcp)—entries specify which hosts are allowed service
- *hosts.deny*(4tcp)—entries specify which hosts are denied service

### 10.1.1 Configuring inetd to use TCP wrappers

---

For a service to use a TCP wrapper, the service must be configured in the */etc/inet/inetd.conf* file to be started through the **in.tcpd** program.

An example of how this is done is the **telnet** server configuration:

```
telnet stream tcp nowait root /usr/sbin/in.tcpd in.telnetd
```

This looks like the normal configuration of a server through **inetd** except that:

- The executable specified is for the **in.tcpd** program.
- The server to be run is specified as an argument to this program.

By default, most services are configured to run through **in.tcpd**, but you should check to verify this is true for the services you are restricting when configuring TCP wrappers.

### 10.1.2 The *hosts.allow* and *hosts.deny* files

---

The */etc/inet/host.allow* and */etc/inet/host.deny* files are used to specify the rules used by **in.tcpd** to implement TCP wrappers. Both files have the same format.

Blank lines or lines that begin with a pound (#) character are ignored. This permits you to insert comments and white space so the tables are easier to read.

All other lines should follow the format:

```
daemon_list : client_list [: options][: ALLOW | DENY]
```

where:

|                    |                                                                                                                                               |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>daemon_list</i> | Lists one or more daemon names.                                                                                                               |
| <i>client_list</i> | Lists one or more host names, host addresses, patterns or wildcards (see below) that will be matched against the client host name or address. |
| <i>options</i>     | Options can be specified. For example, <b>spawn</b> is used to run shell commands.                                                            |
| ALLOW   DENY       | Specifies whether this is an allow or deny rule. If omitted, the default is based on the filename.                                            |

Items in a list are separated by whitespace or commas. The `EXCEPT` token can be used to specify exceptions to list items.

In addition to explicitly specifying the hosts for a rule, the following wildcard patterns can be used:

- Begins with a period ( `.` )—A host name is matched if the last components of its name match the given pattern.

For example, the pattern `“.tue.nl”` matches the host name `wzv.win.tue.nl`.

- Ends with a period ( `.` )—A host address is matched if its first numeric fields match the given pattern.

For example, the pattern `“131.155.”` matches the address of (almost) every host on the network `131.155.*.*`.

- Begins with an at sign ( `@` )—A host name is matched if it is a host member of the specified NIS netgroup. Netgroup matches are not supported for daemon process names or for client user names.
- The form `“n1.n2.n3.n4/m1.m2.m3.m4”` is interpreted as a `“network_address/netmask”` pair. A host address is matched if its network portion, subject to the given netmask, matches the given network address.

For example, the pattern `“131.155.72.0/255.255.254.0”` matches every address in the range `“131.155.72.0”` through `“131.155.73.255.”`

These wildcards can also be used:

|          |                                                                                                                                                                                |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ALL      | The universal wildcard, it matches everything.                                                                                                                                 |
| LOCAL    | Matches any host whose name does not contain a dot character. However, if you are using DNS, all host names include a domain name, so hosts will not match the LOCAL category. |
| UNKNOWN  | Matches any host whose name or address are unknown.                                                                                                                            |
| KNOWN    | Matches any host whose name and address are known.                                                                                                                             |
|          | KNOWN and UNKNOWN should be used with care because host names can be unavailable due to temporary name server problems.                                                        |
| PARANOID | Matches any host whose name does not match its address.                                                                                                                        |
|          | The version of <b>tcpd</b> supplied with UnixWare 7 is built with PARANOID defined, so it drops requests from such clients before looking at the access control files.         |

For example, to design a mostly closed system that only explicitly authorized hosts are permitted to access, edit the */etc/inet/hosts.allow* file:

```
ALL: .local.com
```

```
ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
```

- The first rule permits access from hosts in the local domain.
- The second rule permits access from all hosts in the *foobar.edu* domain (notice the leading dot) with the exception of *terminalserver.foobar.edu*.

All other hosts are restricted by using a trivial */etc/inet/hosts.deny* file:

```
ALL: ALL
```

This denies all services to all hosts unless they were previously permitted access by entries in the *hosts.allow* file.

By contrast, to design a mostly open system that refuses selected hosts, omit the *hosts.allow* file because the default policy is to grant access, which makes the file redundant. The explicitly non-authorized hosts are listed in the */etc/inet/hosts.deny* file:

```
ALL: bad.host.name, .bad.domain
```

```
ALL EXCEPT in.fingerd: other.host.name, .other.domain
```

- The first rule denies the host *bad.host.name* and the *bad.domain* domain access to all services.
- The second rule still permits **finger** requests from the host *other.host.name* and the *other.domain* domain, while denying them access to all other services.

A more complicated example is one of an alarm that sends e-mail to **root** whenever someone tries to contact a service, in this case *tftp*, that they are not allowed to access:

The */etc/inet/hosts.allow* file contains:

```
in.tftpd: LOCAL .my.domain
```

The */etc/inet/hosts.deny* file contains:

```
in.tftpd: ALL: spawn (echo "%h attempted to contact %d" |
/bin/mail -s "wrapper warning" root) &
```

This file permits **tftp**(1tcp) requests from hosts in the local domain. Requests from any other hosts are denied. Instead of transferring the requested file, a mail message is sent to the superuser on the **tftpd** server machine.

Note the use of the character substitutions *%h* and *%d*:

- *%d* expands to the daemon name
- *%h* expands to the host name of the client trying to initiate contact

More character substitutions can be found in the *hosts\_access*(4tcp) manual page.

### **Exercise: Using TCP Wrappers**

---

- Step 1. Create a TCP wrapper that blocks **telnet**(1tcp) connections from all hosts except two specified systems. Do not affect any services other than **telnet**.
- Step 2. Test your wrapper from both the allowed and denied systems to verify that it works.
- Step 3. Examine the `/var/adm/syslog` file to see what information is logged from each connection attempt.
- Step 4. Disable your wrapper when you are finished.

## 10.2 Firewall Basics

---

The term *firewall* comes from barriers that prevent fire from spreading, such as those used in building construction. Likewise, a firewall for a network provides a barrier between LANs and any malicious entities on the Internet.

A firewall is a system that acts as a gateway between networks, usually between your LAN and the Internet. Firewalls do more than pass packets between the networks, they provide a security gate between the two, controlling which packets are allowed to pass.

For a system to act as a firewall:

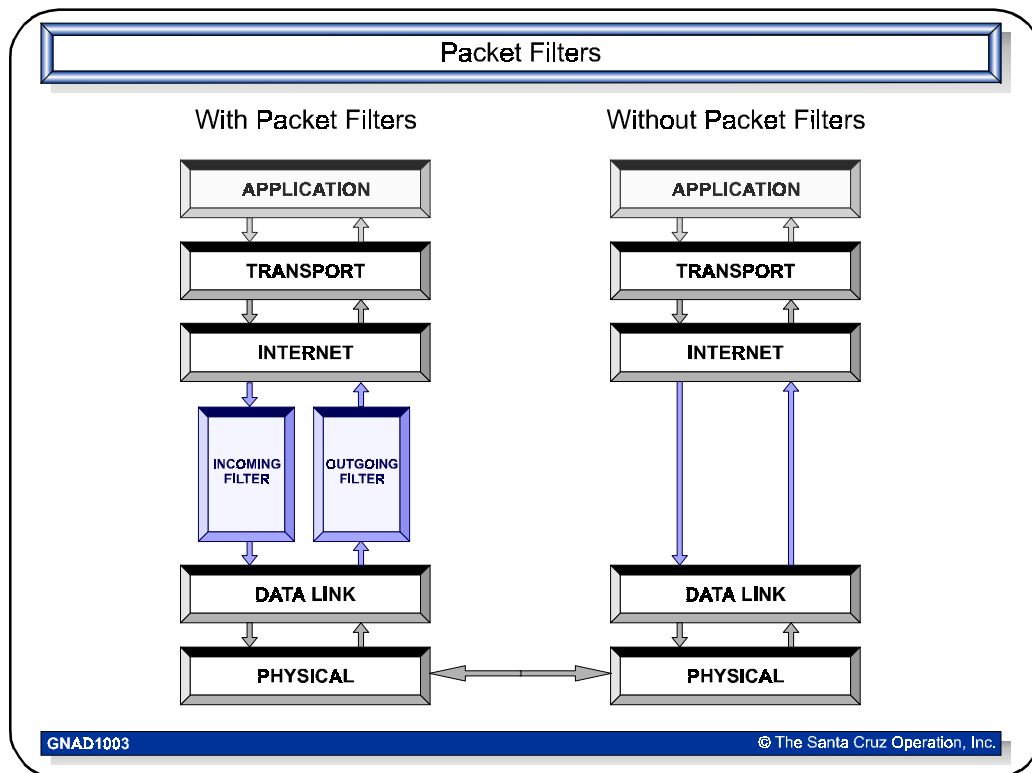
- All traffic from one network to the other must pass through the firewall.
- The machine must monitor traffic and pass only allowed packets.
- The machine itself must be secure.

A properly configured UnixWare 7 system can accomplish the above. To achieve this, the system should be:

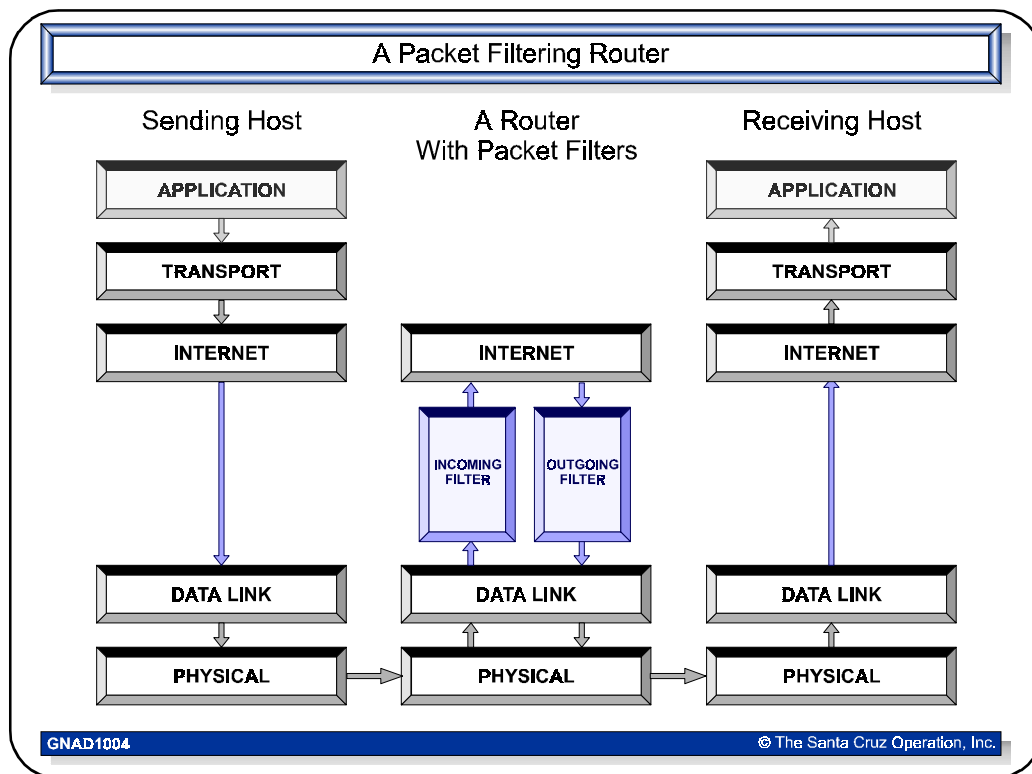
- Set up as the connection between networks.
- Configured with security features such as TCP wrappers, packet filters, and proxy servers.
- Running minimal services and have as few accounts as possible to reduce risk of penetration.

A firewall is not a simple switch that you select on or off. Instead, it is a series of components that you can use to build your security system, like adding bricks to your wall. There are several tools that can be used to build your firewall. Two powerful tools are *packet filters* and *proxy servers*.

AV 10-3



AV 10-4





### 10.2.1 Packet filters

---

A packet filter is a set of rules that control the delivery of packets between the network and Internet layers of the protocol stack. This is shown in AV 10-3.

Packet filters can restrict which packets are allowed to enter or exit through a network interface, effectively providing a barrier to certain services or hosts.

On UnixWare 7 systems, packet filters can be configured on any:

- LAN network interface that uses an MDI network adapter driver
- WAN network interface that uses the PPP serial line protocol provided by UnixWare 7

Packet filters can be configured on a host with a single network interface, though much of the functionality of packet filters for individual hosts can be achieved using the TCP wrappers discussed in Section 10.1, “TCP Wrappers,” on page 10-5.

When packet filters are configured on a host acting as a router between networks, they can:

- Increase the security of your networks against unauthorized access from outside
- Centralize and reduce security administration
- Act as a key component in developing a firewall



An interface can have two associated packet filters:

|          |                                                                |
|----------|----------------------------------------------------------------|
| Incoming | Restricts which packets get into the machine from the network. |
| Outgoing | Restricts which packets get out of the machine to the network. |

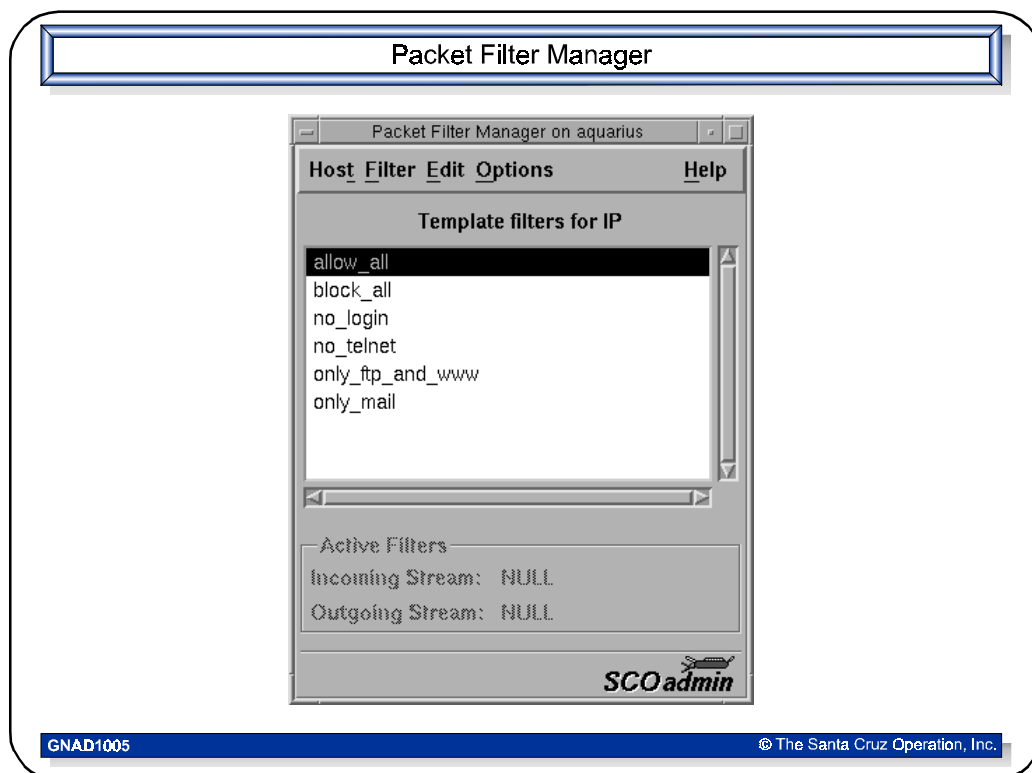
You can also use different types of filters:

|                        |                                                                                                                                                                                                                   |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| An <i>allow</i> filter | <p>Allows all of the packets that match the given rules to pass.</p> <p>Any packets that are not matched by any rule specified are not passed on to the next level of the IP stack (the packets are dropped).</p> |
| A <i>deny</i> filter   | <p>Drops packets that match the given rules.</p> <p>Any packets that are not matched by any rule specified are passed to the next level of the IP stack.</p>                                                      |

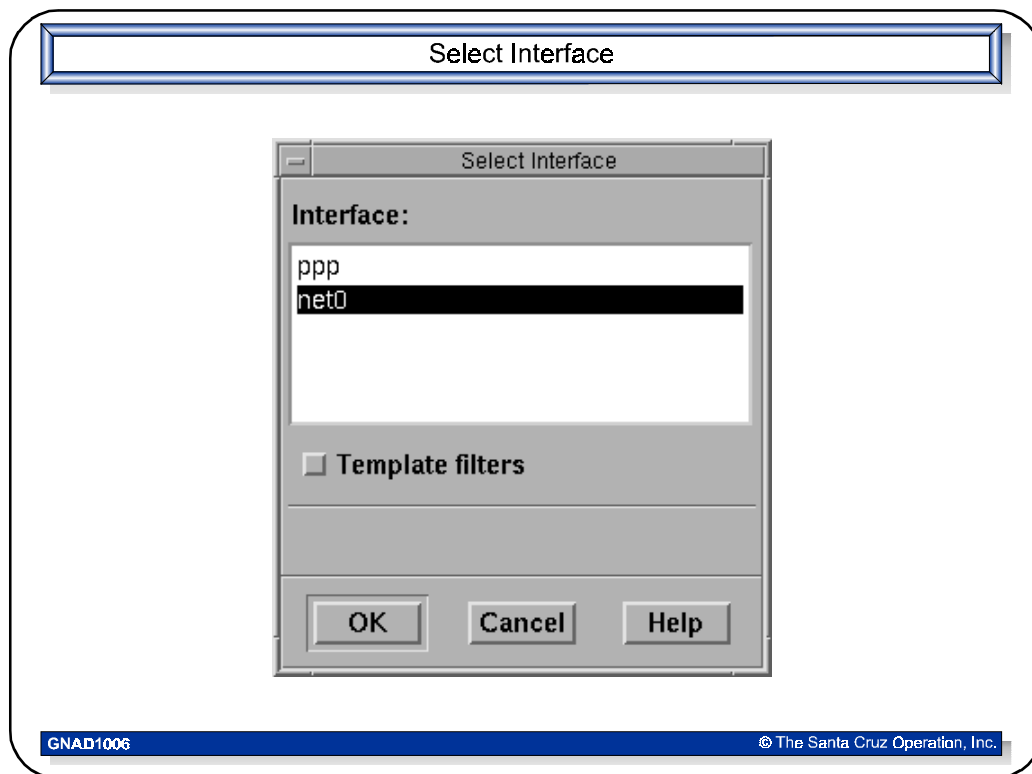
Packet filters are made up of one or more *elements*. An element is an ordered set of rules.

In a packet filter, all elements are of the same type, either allow or deny. For instance, in an allow filter, all the elements are allow elements.

AV 10-5



AV 10-6



### 10.2.1.1 Adding a new packet filter

---

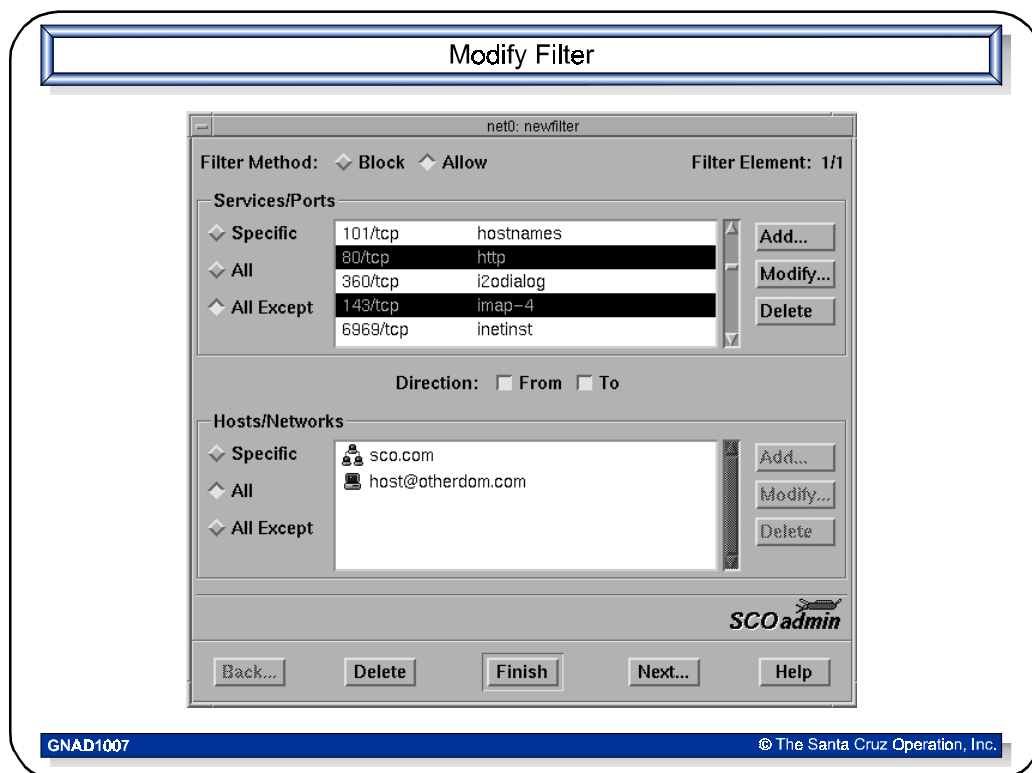
Packet filters for network interfaces can be created, edited, enabled, and disabled using the **Packet Filter Manager**.

To create a packet filter using the **Packet Filter Manager**:

- Step 1. Run **scoadmin(1M)**.
- Step 2. Select **Networking** ⇒ **Packet Filter Manager**.
- Step 3. Select **Filter** ⇒ **Interface**.
- Step 4. Select the interface to filter. Toggle "Template filters" to also display a list of available filters, as shown in AV 10-5.
- Step 5. Click on **OK**.
- Step 6. Select **Edit** ⇒ **Add**.
- Step 7. Enter a name for the filter in the "New Filter" field in the pop-up window.
- Step 8. Click on **OK**.

Now, you can define the filter elements as discussed in Section 10.2.1.2, "Defining packet filter elements," on page 10-19.

## AV 10-7



### 10.2.1.2 Defining packet filter elements

---

A packet filter consists of a series of rules. These rules are referred to as *elements* in the **Packet Filter Manager**.

After completing the steps in Section 10.2.1.1, “Adding a new packet filter,” configure this packet filter using the form shown in AV 10-7:

Step 1. Select the “Filter Method” to use. All elements in a filter must use the same method. This configures whether the filter rules allow packets through or blocks packets from passing. You can specify:

- **Allow**—Only packets you explicitly allow are passed; any packets not matching a rule are blocked.
- **Block**—Only packets you explicitly block are stopped; any packets not matching a rule are allowed.

Step 2. Specify which “Services/Ports” are covered by this filter. You can select from the options:

- **Specific**—This element only covers the selected services.
- **All**—This element covers all services.
- **All Except**—This element covers every service except for those selected.

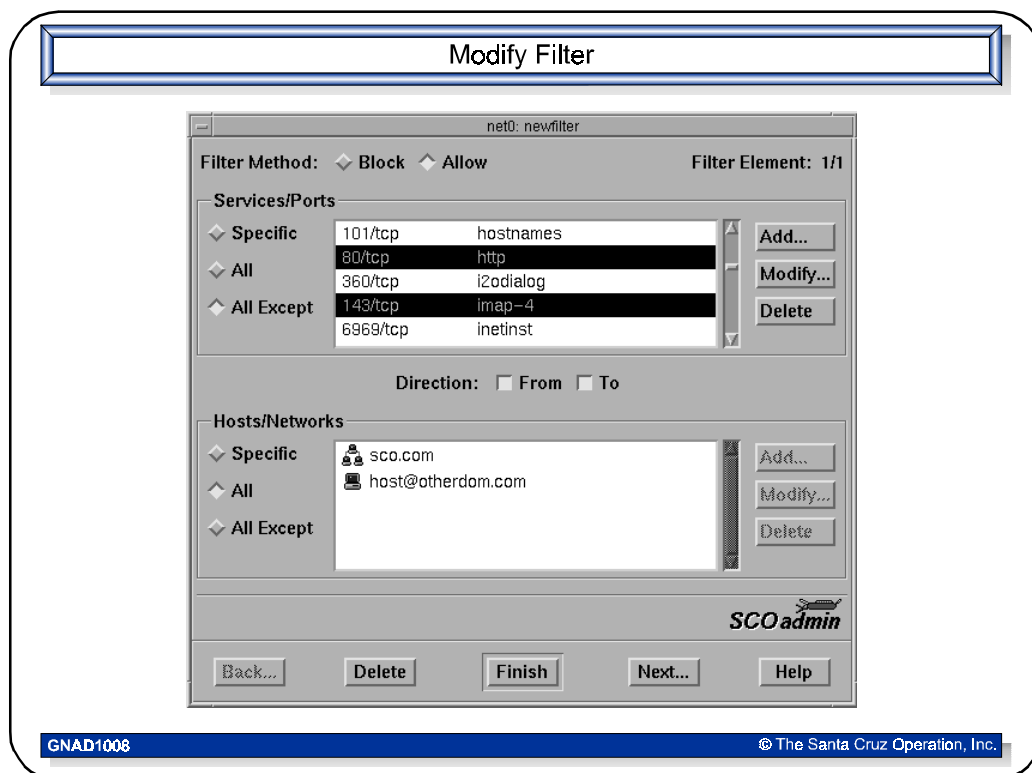
Unless you select **All**, you must also specify services in the selection window. Multiple items can be selected by pressing the  key while selecting the items.

Note that items are selected merely by highlighting them. The **add**, **modify**, and **remove** buttons are for modifying the list of available services.

If the applicable services do not appear in the list, they can be added using the **Add** button.

Services can also be modified or deleted using the **Modify** and **Delete** buttons. Services that appear in the `/etc/services` file or ones in use by other filters cannot be deleted or modified.

## AV 10-8





- Step 3. Choose the “Direction” to which this element applies. This specifies whether the filter applies to packets coming **From** or going **To** the hosts or networks specified in the following area. You can select both if you want the rule to apply to traffic in both directions.
- Step 4. Specify to which “Hosts/Networks” this filter element applies. You can select from these options:
- **Specific**—This element only applies to the selected hosts or networks.
  - **All**—This element applies to any host or network.
  - **All Except**—This element applies to every host or network except those selected.
- Step 5. Click on one of the following when you complete this element:
- **Back**— Go to the previous filter element in this filter definition.
  - **Delete**—Delete this filter element. Deleting the last element resets the element to its default selections.
  - **Finish**—Exits this window. You are prompted to save the changes.
  - **Next**—Moves to the next element in the filter definition, or adds a new element if the current element is the last one. If all packets are covered by the current element, **Next** is grayed out.

### 10.2.1.3 Loading and unloading a packet filter

---

When you have finished defining the elements in a packet filter, load the filter to take effect. To load a packet filter:

- Step 1.        Select the filter in the Packet Filter Manager window.
- Step 2.        Select **Filter** ⇒ **Load**.
- Step 3.        Select the “Direction” in which you want to load the filter. You can apply the filter to **Incoming**, **Outgoing**, or both directions.  
  
                 Only one filter may be applied to each direction for a particular interface.
- Step 4.        Click on **OK**.

To unload a packet filter:

- Step 1.        Select the filter in the Packet Filter Manager window.
- Step 2.        Select **Filter** ⇒ **Unload**.
- Step 3.        Select the “Direction” from which you want to remove the filter. You can remove the filter from the **Incoming**, **Outgoing**, or both directions.
- Step 4.        Click on **OK**.

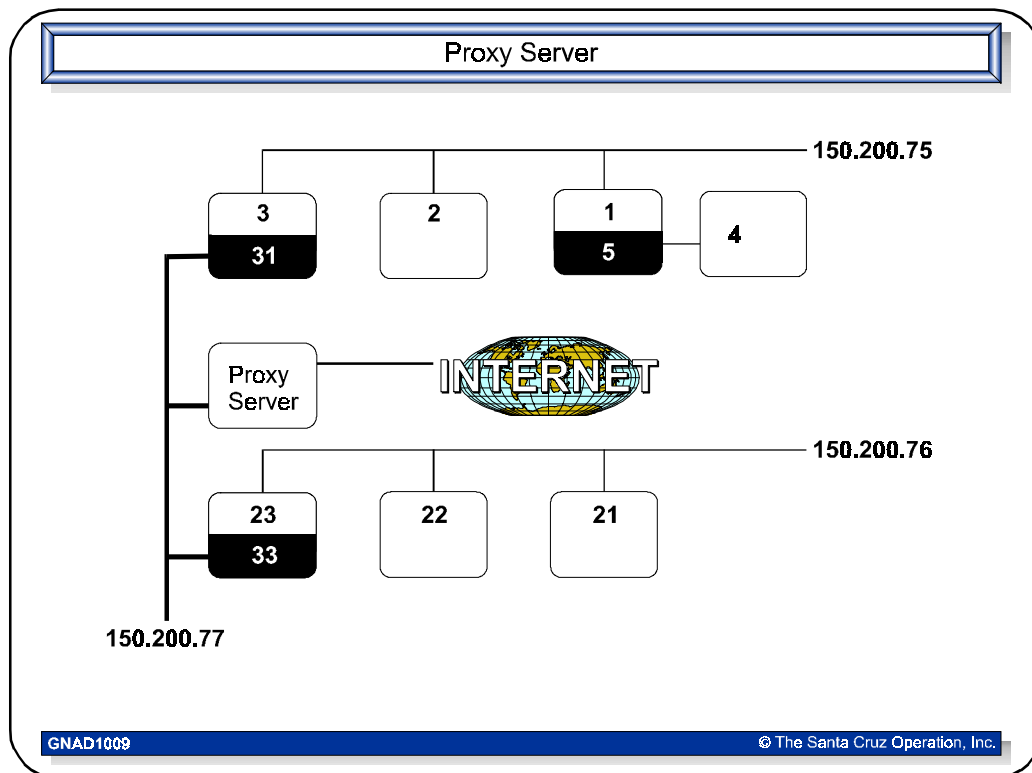
**Exercise: Implementing a packet filter**

---

- Step 1.        Implement a packet filter on incoming packets on you network interface to prevent **telnet** access from all but one trusted host.
- Step 2.        Test your filter from both the trusted and non-trusted hosts.
- Step 3.        Disable your filter when you are finished.

If you have a UnixWare 7 system configured as a router, you can perform this lab on the router to see that it blocks packets that are not addressed to the router itself. This shows that packet filters are significantly more powerful than TCP wrappers.

AV 10-9



## 10.3 Proxy Servers

---

Additional security can be implemented on networks using proxy servers. Instead of passing the packets through the gateway for a service, a router can block the packets. It can do this by refusing to route any packets, or through the use of a packet filter. If your local machines require access to specific services outside the network, a proxy server can then be configured on the connecting machine.

A proxy server receives requests from clients and acts on their behalf on the outside network. This is analogous to giving someone your proxy to act on your behalf in legal matters.

Most proxy daemons only proxy specific services. For example, if you want a machine inside an isolated network to have access to web pages, the machine connected to the outside must be running a proxy server that will proxy the required protocols HTTP, FTP, and gopher.

The client applications must be configured to make use of this proxy server. After this is done, these hosts are able to use the proxied service, but are still unable to otherwise communicate with the outside network.

An additional benefit to using a proxy server is the ability to use non-registered IP addresses on a local area network hidden behind a proxy. This can be done because there are no packets coming directly from the LAN. All the packets sent on to the Internet appear to be coming from the proxy machine.

Proxy servers are commonly run exclusively for web browsers. They proxy the HTTP, HTTPS, FTP, and Gopher protocols for properly-configured browsers. Some proxy servers available on UnixWare 7 for this purpose are:

|            |                                                                                 |
|------------|---------------------------------------------------------------------------------|
| Squid      | Available on the Skunkware CD-ROM                                               |
| CERN httpd | Source code freely available on the web; must be compiled to run on UnixWare 7. |

Proxy servers can also provide performance benefits because most can cache web pages and graphics on the proxy server system. Subsequent requests for the same page can be satisfied immediately, without contacting the original web server across the network.

### 10.3.1 Using a proxy server

---

To use a proxy server, you must configure your web browser. When configured, the browser sends all requests for web pages to the proxy server on the specified port, instead of routing them directly to the desired web server. Most, but not all, browsers support the use of a proxy server.

Configuring your browser to use a proxy server will vary, depending on which particular browser you are using. However, the process is similar and this general procedure can be used as a guide. For details on your browser software, see your online help system.

To configure a Windows-based browser to use the proxy server:

1. Locate your browser's **Network** configuration screen. This is often accessed through **Preferences** or **Options** screens.
2. Locate the screen or area for configuring **Proxies**.
3. Locate the field for configuring the proxy server. Enter this URL:

`http://proxyserver:8080`

where *proxyserver* is the host name or IP address of the proxy server system and 8080 is the port.

To configure a UNIX-based browser, such as Lynx or NCSA Mosaic, set the proxy-related variables, as in this example Mosaic configuration file `/usr/lib/X11/app-defaults/Mosaic`:

```
Mosaic*httpProxy: http://proxyserver:8080/
Mosaic*ftpProxy: http://proxyserver:8080/
Mosaic*gopherProxy: http://proxyserver:8080/
```

where *proxyserver* is the host name or IP address of the proxy server system, and 8080 is the port number used by the proxy server.

Note that these will not work if you omit the trailing slash in the URLs.

## 10.4 Packet Security

---

Standard TCP/IP communications are devised to effectively and reliably transmit packets over a network. Unfortunately the TCP/IP protocol has little built in security and can fall prey to a variety of attacks. These attacks can include:

- Eavesdropping or packet sniffing—It is possible for someone to *listen in* on your communications. This is undesirable, especially when the information is of a guarded nature, such as credit card numbers.
- Modification—The packets sent from one system can be intercepted and replaced with new packets to deliver modified information.
- Impersonation—It is possible for one computer or user to pretend to be:
  - Someone they are not. They can try to take over the identity of an existing system or user on the net. This is called *spoofing*.
  - Something they are not. They might not be using another identity but simply providing a false identity of their own. For example, a site might pose as a retail order site collecting orders and credit card numbers but never actually shipping anything.





### 10.4.1 Packet sniffing

---

If you are carrying on a conversation in a crowded room, you would not expect the conversation to be confidential and might be concerned about eavesdroppers. Similarly, you might have concerns about information transmitted using the IP protocol. The IP protocol is designed for the effective transmission of information but takes no precautions to protect against eavesdropping or as it is commonly referred to in this context, *packet sniffing*.

When information is sent between systems on an Ethernet, all hosts on the Ethernet can see these packets. Similarly, when sending information over the Internet, all the hosts in the path between the systems have access to the packets and the data they contain.

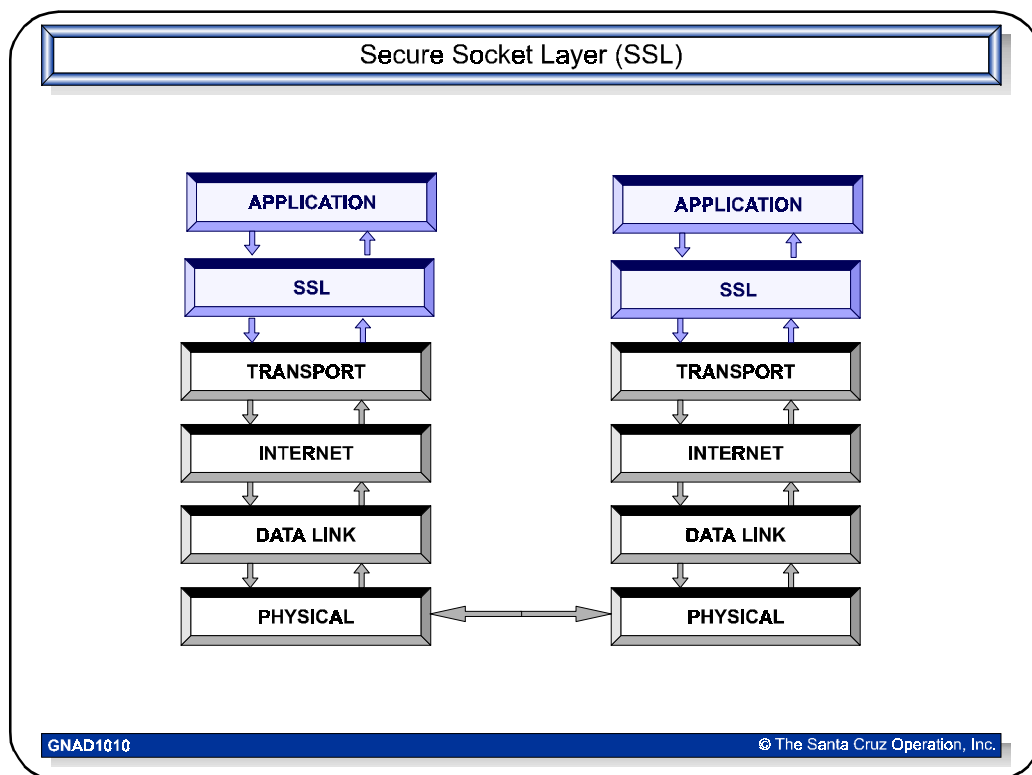
Normally, a host ignores any packet that is not addressed to the host. However, a host can be configured to listen for these packets. This can present a danger especially with packets containing information such as passwords or credit card numbers.

Despite these dangers, packet sniffers can also have legitimate uses:

- Monitoring network traffic for security purposes
- Diagnosis of network problems
- Measuring network traffic

If you have a packet sniffer running, watch a **telnet** session to see how easy it is to snoop passwords sent by **telnet**.

AV 10-10



## 10.4.2 The Secure Socket Layer (SSL)

---

The Secure Socket Layer (SSL) is a security protocol that prevents eavesdropping, tampering, or message forgery and is commonly used as a component of an electronic commerce solution.

SSL was developed to overcome the security shortcomings of TCP/IP. Since its inception, it has become a widely accepted standard used by many networking products. It is the basis for the Transport Layer Security (TLS) protocol under development by the Internet Engineering Task Force (IETF).

SSL provides several features that safeguard against the threats mentioned above, including:

|                 |                                                                                                                                                                                                                                                  |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Confidentiality | By encrypting messages, SSL ensures that message intercepted enroute are essentially unreadable.                                                                                                                                                 |
| Integrity       | Encryption ensures integrity; if the message is modified in any way, it will not decrypt properly.                                                                                                                                               |
| Authentication  | SSL uses certificates that are issued by various authorities willing to attest that you are who you claim to be. Providing you trust the authority that issued the certificate, you can be reasonably sure of the remote host or users identity. |

Additionally, the Secure Socket Layer is:

- Implemented as a layer in the network stack between the application and transport layers, as shown in AV 10-10
- Currently at version 3.0; Version 3.0 is backward-compatible to version 2.0
- Certificate based; SSL V3.0 uses X.509 V3 certificates to ensure authenticity
- Used by Web servers when a URL specifies "https" instead of "http"

For a detailed description of the steps necessary to enable SSL on your web server or for more information on SSL operation, refer to **scohelp** ⇒ **Internet and Intranet** ⇒ **Netscape** ⇒ **Netscape Administrator's Guide** ⇒ **Chapter 7 Encryption and SSL**.

## 10.5 Lesson Summary

---

---

This lesson described dangers imposed by connecting your systems to untrusted networks. It provided an overview of security precautions that you can implement on your systems and networks to guard against network-related security threats. An example of packet filters, a major component in building a firewall, was presented.

The tools available to enhance the security of web servers, including proxy servers and the Secure Sockets Layer protocol were also described.

## Review questions

---

1. Which files are used to configure the use of TCP wrappers to provide security for a network service provided by the system?
2. What qualifies a machine as a firewall?
3. What are the advantages of using a packet filter over using TCP wrappers?
4. Why would you run a proxy server on a firewall?

5. Can all hosts on the Internet sniff packets sent or received by your system? Explain your answer.
  
6. What measures can you implement to guard against your packets being sniffed?

## 10.6 Relevant Commands and Files

---

---

The following commands and files were presented in this lesson.

### Relevant commands

---

|                         |                                                   |
|-------------------------|---------------------------------------------------|
| <b>finger</b> (1tcp)    | Displays information about local and remote users |
| <b>ftp</b> (1tcp)       | Invokes the Internet file transfer program        |
| <b>ifconfig</b> (1Mtcp) | Configures network interface parameters           |
| <b>inconfig</b> (1Mtcp) | Configures TCP/IP-related kernel parameters       |
| <b>telnet</b> (1tcp)    | Invokes the remote login program                  |

### Relevant files

---

|                        |                                                                                     |
|------------------------|-------------------------------------------------------------------------------------|
| <i>/etc/inetd.conf</i> | Contains configuration information for <b>inetd</b> ; see <i>inetd.conf</i> (4tcp). |
|------------------------|-------------------------------------------------------------------------------------|

## 10.7 Answers to Review Questions

---

1. The *inetd.conf*, *hosts.allow*, and *host.deny* files are used to configure the use of TCP wrappers.
2. The qualifications for a machine to be a firewall are that all the packets between the networks must pass through the machine, the machine must be restricting packets, and must itself be secure.
3. The advantages of using a packet filter over using TCP wrappers include: TCP wrappers restrict access to services on the host, packet filters restrict *all* packets passing through an interface and therefore can restrict access to an entire network when implemented on a router.
4. You would run a proxy server over a firewall because a proxy server allows access to services on the Internet through your router without compromising its security. Proxy servers can also provide performance benefits such as providing a centralized cache.
5. All hosts on the Internet cannot sniff packets sent or received by your system. Only those hosts on networks that the packet passes through would be able to sniff the packet. This means that local traffic which is transmitted solely on a LAN could only be seen by other hosts on the LAN. Packets traveling over the Internet can take unpredictable routes to their destination and any hosts on the traversed network segments would have the ability to sniff the packet.
6. The measures that you can implement to guard against your packets being sniffed include implementing subnets to keep more secure data on restricted networks. For packets that require security but are traversing untrusted network connections, encryption such as SSL may be a prudent precaution. Another option is to use a network switch to restrict the packets that are sent to each host.



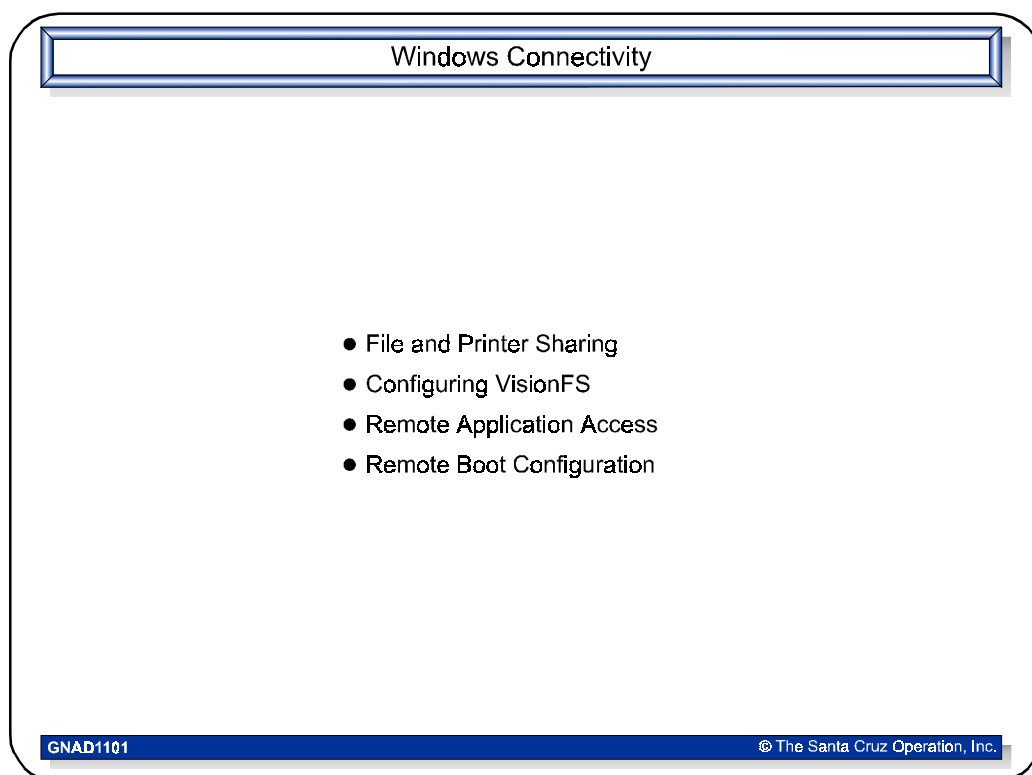


---

# ***11***

## ***Windows Connectivity***

AV 11-1



**Lesson 11**

## **Windows Connectivity**

### **Introduction**

---

In addition to communication between UNIX<sup>®</sup> systems, we often need to communicate between UNIX and other operating systems. This lesson focuses on strategies and tools for providing services to Microsoft Windows based computers on your network. The strategies discussed can also be used with a wide range of operating systems beyond the Windows family.

The primary services covered are file sharing, printer sharing, remote application access, and remote boot configuration. SCO VisionFS<sup>™</sup>, a tool included with UnixWare 7<sup>®</sup>, is used to illustrate file and print sharing. This lesson introduces VisionFS in addition to summarizing other options that provide similar services.

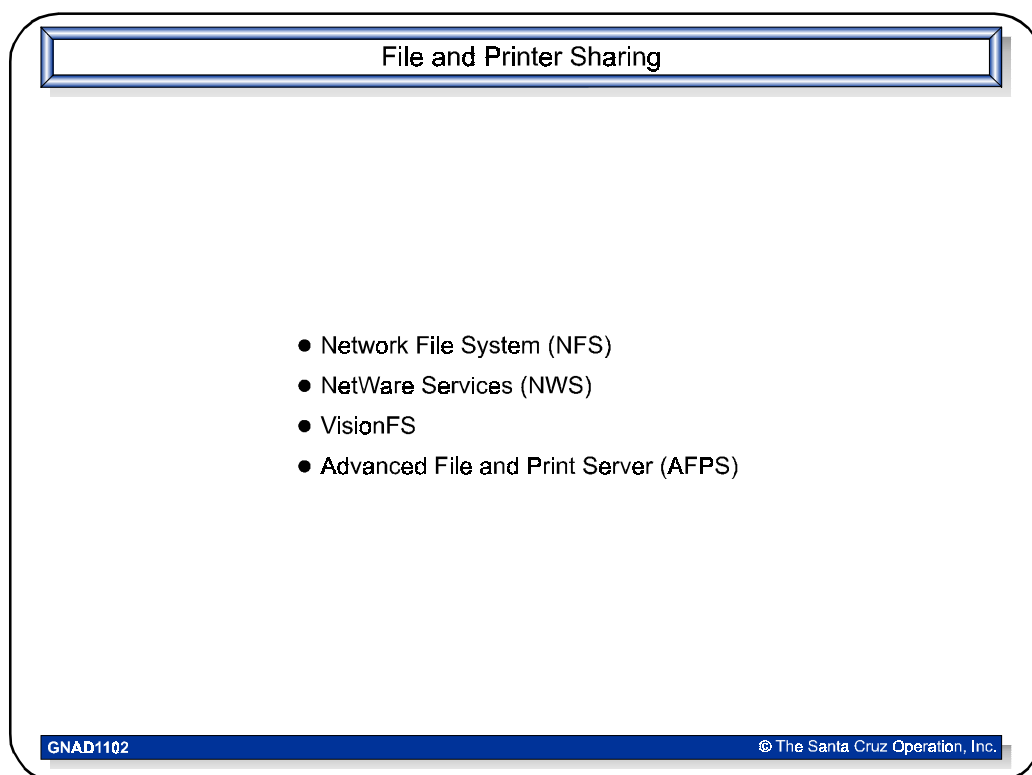
### **Objectives**

---

This lesson presents options for providing connectivity to Windows and other client operating systems. By the end of this lesson, you will be able to:

- Describe four types of Windows connectivity services
- Describe four options for providing file and printer sharing for Windows
- Install and configure VisionFS to provide file and printer sharing
- Describe four options for providing remote application access for Windows
- Use SCO TermLite<sup>™</sup> to run UNIX commands from a Windows client
- Install and access the Webtop to run server applications from any client
- Use DHCP to provide remote boot configuration services

AV 11-2



## **11.1 File and Printer Sharing**

---

Your network can consist of servers running UNIX and client systems running other operating systems. Files can be stored on the UNIX server and made available to your client systems. In addition, your client users can print to printers attached to your UNIX system. UnixWare 7 provides several alternatives for both tasks.

To provide file and printer sharing:

- Select a network protocol for exchanging the relevant data.
- Install client-side software that implements the selected protocol(s).
- Install server-side software that implements the selected protocol(s).

Four approaches for providing file and printer sharing from a UnixWare 7 server include:

- Network File System (NFS®) (file sharing only)
- NetWare® Services (NWS)
- VisionFS
- SCO Advanced File and Print Server (AFPS)

### 11.1.1 NFS

---

UnixWare 7 includes both server and client support for the NFS file sharing protocol. NFS does not provide printer sharing.

Advantages of using NFS to provide file sharing include:

- NFS server software is included with almost every version of UNIX. This allows your NFS clients to easily connect to a wide variety of servers.
- NFS server software is available for a wide variety of non-UNIX operating systems. This allows your NFS clients to connect to an even wider variety of servers without using a different interface or protocol for each one.
- NFS client software is included with almost every version of UNIX. This lets you access your shared files from other UNIX machines.
- NFS client software is available for a wide variety of non-UNIX operating systems. This allows easy access to your files independent of which client you are using.

Disadvantages of using NFS include:

- NFS client software is not included with Windows 95, Windows for Workgroups, or Windows NT. To use NFS with these operating systems, you will need to purchase, install, and configure software for every client machine.
- The NFS client software you install might not provide the same interface for browsing file and directories that Windows uses by default. If not, this will require training all of your users with a new interface.
- NFS is designed for file sharing between UNIX systems. It does not by itself perform mapping between Windows and UNIX filenames or file permissions. Although, a particular implementation can provide these translations.

In summary, NFS is a good choice if you have UNIX clients or a wide variety of client and server operating systems. However, requiring client-side software installation makes it inconvenient to use with many Windows clients.

### **11.1.2 NetWare Services**

---

Another option is to use the NetWare Services (NWS) package to provide Novell® NetWare style file and print sharing to NetWare client systems. NWS allows UnixWare to appear as if it were a NetWare 4.11 server on the network. This allows NetWare client machines to access UnixWare 7 as a NetWare server.

Advantages of using NWS to provide file and printer sharing include:

- If you already have NetWare servers, NWS allows you to use UnixWare 7 to serve as an additional file server, without altering client setup. Your NetWare client machines can access NetWare and UnixWare 7 servers interchangeably.
- NetWare client software is included with Windows 95 and Windows NT. You do not have to purchase and install additional software for every client.
- NetWare client software is widely available for DOS, making it easy to provide file sharing to DOS users.
- NWS supports the Novell NDS directory tree.

Disadvantages of using NWS include:

- When only sharing files between UnixWare 7 and Windows, NWS can be complicated. It requires familiarity with NetWare concepts and techniques, even though you are not using NetWare servers.

Troubleshooting can be difficult because you need to understand how permissions, file locks, caching, etc., are translated from Windows to NetWare to UnixWare 7.

- Client user licenses for NWS are not included with UnixWare 7 and must be purchased separately.

In summary, NWS is a good choice if you already have NetWare client or server systems in place and want to add a UnixWare 7 file server without reconfiguring your clients. If you do not already have an investment in NetWare networking, and your clients are using Windows, then it is probably not a cost-effective solution.

### **11.1.3 VisionFS**

---

A simple and economical option for providing file and printer sharing for Windows clients is VisionFS. VisionFS uses the Common Internet File System (CIFS) protocol to share files and printers. CIFS is the native protocol used by Windows 95 and other Windows versions to share files and printers.

Advantages of using VisionFS to provide file and printer sharing include:

- The CIFS client software is built into most versions of Windows, including Windows for Workgroups, Windows 95, and Windows NT. There is no client software to purchase. Often there is no client software to install because it has been installed already for Windows-to-Windows file sharing.
- VisionFS runs on many different versions of UNIX, eliminating the need to learn a new product for each different UNIX server.
- VisionFS is designed to share files and printers with Windows clients handling Windows users, passwords, filenames, permissions, etc.
- Because users are probably already familiar with the Windows interfaces used to access CIFS shared files and printers, there is little or no user training required.
- VisionFS is relatively simple to configure and administer.

Disadvantages of using VisionFS include:

- It might be difficult to access your shared files and printers from UNIX clients or from operating systems where CIFS client software does not exist.
- You are limited to having no more simultaneous clients than you have UnixWare 7 user licenses. NFS, on the other hand, does not license or limit client access.

In summary, VisionFS is probably the best choice if the client systems already support CIFS, as do Windows 95, Windows NT, and Windows for Workgroups.



## 11.2 Installing and Configuring VisionFS

---

The VisionFS package can be installed during Initial System Load (ISL). Use this command to determine if VisionFS is already installed on your system:

**pkginfo visionfs**

If VisionFS is installed, similar output displays (perhaps with a different version number):

```
application visionfs SCO VisionFS 3.0
```

The VisionFS package is found on the Optional Services CD-ROM.

During the installation of VisionFS, you are prompted to change the default configuration. On most systems the default configuration is sufficient. If you later need to change any of these settings, you can use the VisionFS **Profile Editor** or the command:

**/usr/vision/bin/visionfs setup**

### 11.2.1 VisionFS passwords

---

After installation, passwords must be set up for existing UNIX user accounts. VisionFS includes three ways to authenticate users:

|                |                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------|
| UNIX           | The UNIX password for the user account is used to authenticate the user.                                           |
| Another server | Logon is authenticated by another system, such as a Windows NT Server.                                             |
| VisionFS       | A separate password database can be set up by VisionFS to store encrypted passwords. This is the default behavior. |

VisionFS stores passwords using the same encryption mechanism as newer releases of Windows 95 and Windows 98. Earlier VisionFS releases used UNIX passwords by default, which allowed unencrypted passwords to be transmitted over the network. Current Windows workstations do not support unencrypted passwords.

Before installing VisionFS, determine whether you need to configure encrypted passwords. Then run the password wizard with the applicable option. See Section 11.2.3, “Running the password wizard,” on page 11-11.

## 11.2.2 Installing VisionFS

---

To install VisionFS and TermLite, follow these steps:

- Step 1. Insert the UnixWare 7 Optional Services CD-ROM.
- Step 2. Start **SCOadmin**.
- Step 3. Select **Software\_Management** ⇒ **Application Installer**.
- Step 4. Select the CD-ROM device from the **Install From** pull-down menu.
- Step 5. Click on **Update View**.
- Step 6. Select **VisionFS3**.
- Step 7. Click on **Install**.
- Step 8. Tab to **Apply** in the **Add Application** window to install both VisionFS and TermLite. Press  to continue.
- Step 9. Enter **y** to accept the default installation settings. Press  to continue when the summary screen displays a successful installation.  
  
After it is installed, a **VisionFS3** icon appears in the lower window of the **Application Installer**.
- Step 10. Select **Actions** ⇒ **Exit** to exit the **Application Installer**.

### 11.2.3 Running the password wizard

---

After VisionFS is installed, run the password wizard to set passwords for existing UNIX accounts:

Step 1. Start the password wizard. Enter:

```
/usr/vision/bin/visionfs password --wizard
```

Step 2. Press **ENTER** at the “Continue or Quit?” prompt.

Step 3. Select a method for choosing passwords. NOTE: This does not affect the UNIX password. It creates a VisionFS password for users accessing the server from the network. Choose from:

- |   |                                                                                                                                              |
|---|----------------------------------------------------------------------------------------------------------------------------------------------|
| B | Sets each user’s VisionFS password to be blank                                                                                               |
| F | Allows you to enter a single password which will be used for all VisionFS accounts                                                           |
| R | Generates a unique random password for each VisionFS account                                                                                 |
| U | Sets the VisionFS password to be the same as the UNIX user name. NOTE: This does not set the VisionFS password to the UNIX <i>password</i> . |

Step 4. Enter **n****ENTER** to accept the passwords generated by the wizard.

Otherwise, enter **y****ENTER** to manually make *additional* changes to the VisionFS password database. This starts **vi**(1) to modify */usr/vision/vfsprofile/password.default*.

When finished, enter the **vi** command **:wq**.

Step 5. Enter **r****ENTER** to overwrite (save the changes made) to the password database. Generally, you will want to overwrite the database, since that is the purpose of the wizard.

- Step 6. Enter **n**  to send mail to users notifying them of password changes. If you choose to send mail, you must respond to a series of prompts to configure the message and other mail information.
- Step 7. After noting or communicating password changes, remove the `/usr/vision/vfsprofile/password.default` file to prevent non-authorized users from obtaining VisionFS password information.

The VisionFS server can now be started. Enter:

```
/usr/vision/bin/visionfs start
```

To verify that the server is running, enter:

```
/usr/vision/bin/visionfs status
```

VisionFS is provided with a full license for Business, Data Center, and Enterprise editions of UnixWare 7.1. Licenses can be separately purchased for Base and Messaging editions.

### 11.2.4 Adding users and changing passwords

---

If you add UNIX accounts after running the VisionFS password wizard, they are not automatically added to the VisionFS password database. You must run the password command to add users individually before the new users can access VisionFS shares from a client.

For example, to set the password for a new UNIX account, **bar**, enter:

```
/usr/vision/bin/visionfs password --add bar bar
```

This sets the VisionFS password to be the same as the UNIX account name, **bar**.

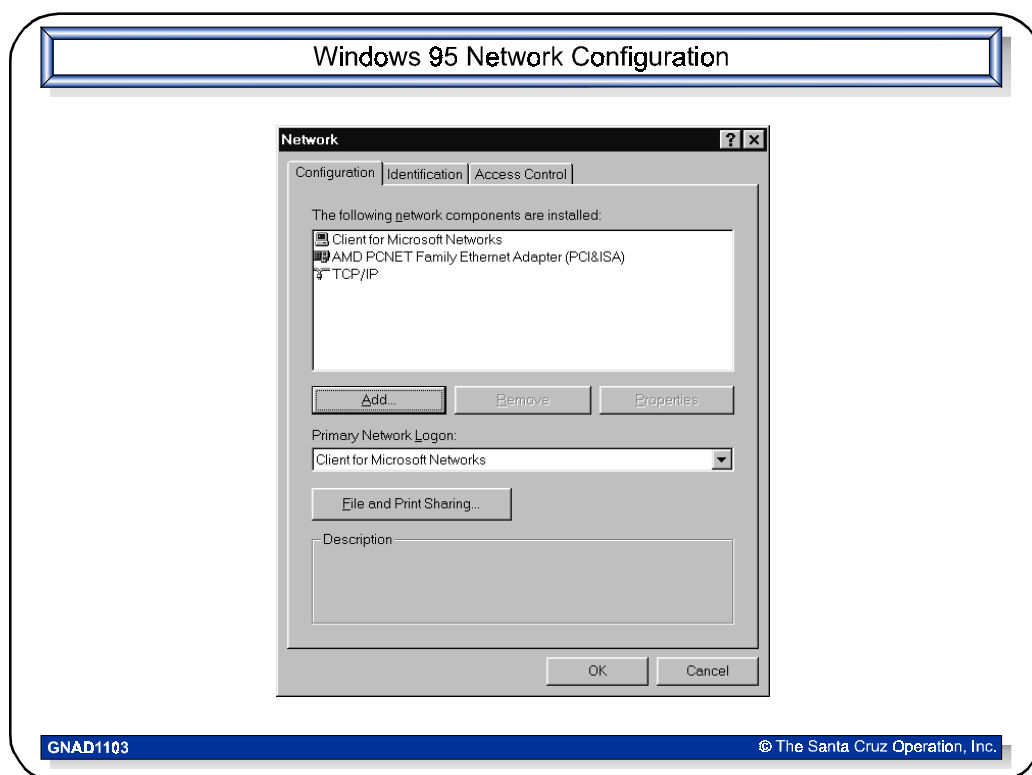
The VisionFS password command is also used to update the VisionFS password database when the UNIX password is changed for a user account. If the UNIX password is changed, and the VisionFS password database is not updated, the user will be prompted for a password whenever accessing a shared resource.

### **Exercise: Installing VisionFS**

---

- Step 1. Determine if VisionFS is installed on your system.
- Step 2. If it is not present, install VisionFS on your system. Accept the default configuration options.
- Step 3. Run the password wizard with these selections:
- Select the **u** option to use UNIX account names as VisionFS passwords.
  - Select **n** when prompted to make additional changes.
  - Select **r** to overwrite the database (saving the changes).
  - Select **n** to bypass sending mail.
- Step 4. Start the server.

AV 11-3



## 11.2.5 Windows network configuration

---

Although all of the software needed to work with VisionFS is included with each Windows workstation (releases later than Windows 3.1), it might not be installed or correctly configured on every client system.

There are four types of networking components that can be configured on a Windows workstation:

|          |                                                             |
|----------|-------------------------------------------------------------|
| Adapter  | Adds a driver for the network adapter hardware              |
| Protocol | Configures protocol settings, such as IP address for TCP/IP |
| Client   | Enables access to server resources                          |
| Service  | Allows files on this computer to be shared                  |

You must configure the Adapter, Protocol, and Client components to access a VisionFS server.

This basic procedure outlines the configuration of a Windows workstation as a VisionFS client. The specific steps used depend on the version of Windows you are using.

- Step 1.        Install a network adapter into the workstation, if not already present.
- Step 2.        Boot Windows and log in as any Windows user (if prompted).
- Your network adapter might be detected during system boot. If so, the adapter driver and several protocols are installed automatically and you can skip steps 4 and 5.
- Step 3.        Access network configuration screens by way of the **Network** icon in the **Control Panel**. Any installed networking components will be listed.

- Step 4. Add the Adapter component to install a driver for your network card.
- Step 5. Add the Microsoft TCP/IP Protocol component and specify an IP address and subnet mask appropriate for your network. Note that this workstation should be on the same subnet as the server.
- Step 6. Add the Client component, Client for Microsoft Networks.
- Step 7. Add the Service component, File and Print Share.
- Step 8. Save the configuration and reboot the system.
- Step 9. Log onto the network using a UNIX account name and the corresponding VisionFS password.

You can now access your VisionFS server by double-clicking on the **Network Neighborhood** icon on the desktop or mapping a network drive using the **File Manager**, depending on the Windows version you are using.

If your VisionFS server is not listed:

- Step 1. Select **Start** ⇨ **Run** from the Windows menu.

- Step 2. In the “Open” field, enter:

`\\servername`

where *servername* is the name of your VisionFS server.

If prompted, enter the VisionFS password for the user name under which you are logged onto the network.

### **Exercise: Configure Windows for VisionFS**

---

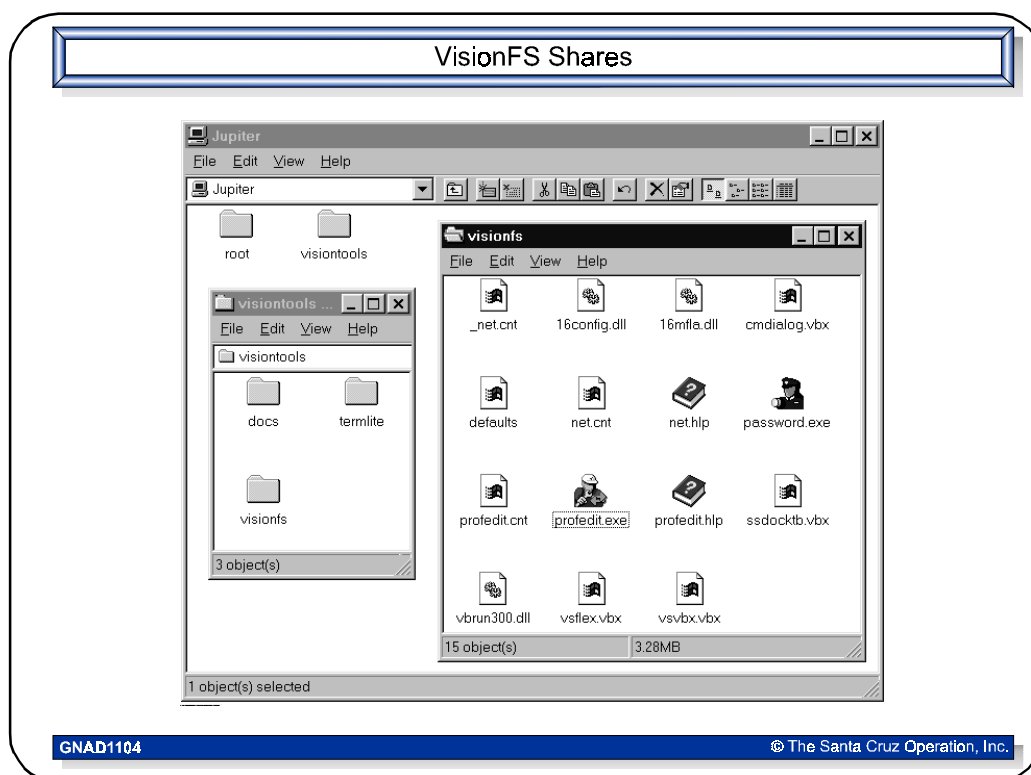
Configure your Windows system so that it can access your VisionFS server. Obtain information such as network interface card model, hardware configuration, and an IP address from your instructor.

You should see the *visiontools* share (possibly among others) on the VisionFS server. You might not yet be able to read or write to any shared directories because we have not discussed users or permissions.



NOTE: There is a problem with some versions of Windows 98 and Windows NT Workstation. When using the **Network Neighborhood** to browse the VisionFS server, you are prompted to enter a password. For information about a solution to this problem, see the problem-solving information on the Optional Services CD-ROM in *./info/visionfs/readme*

AV 11-4



## 11.2.6 VisionFS administration overview

---

Although some VisionFS administration can be performed from the UNIX command line, most tasks are best performed from the *VisionFS Profile Editor*. The **Profile Editor** is a Windows application that can be used to perform remote configuration of your VisionFS server.

Only a *VisionFS administrator* has permission to run the **Profile Editor**. By default, there is one VisionFS administrator, with the user name of **root**, on both the Windows and UNIX systems.

The two **root** user accounts (UNIX and Windows) should use the same password. If the passwords are not the same, VisionFS prompts you for the UNIX **root** password when you try to access shared directories or printers.

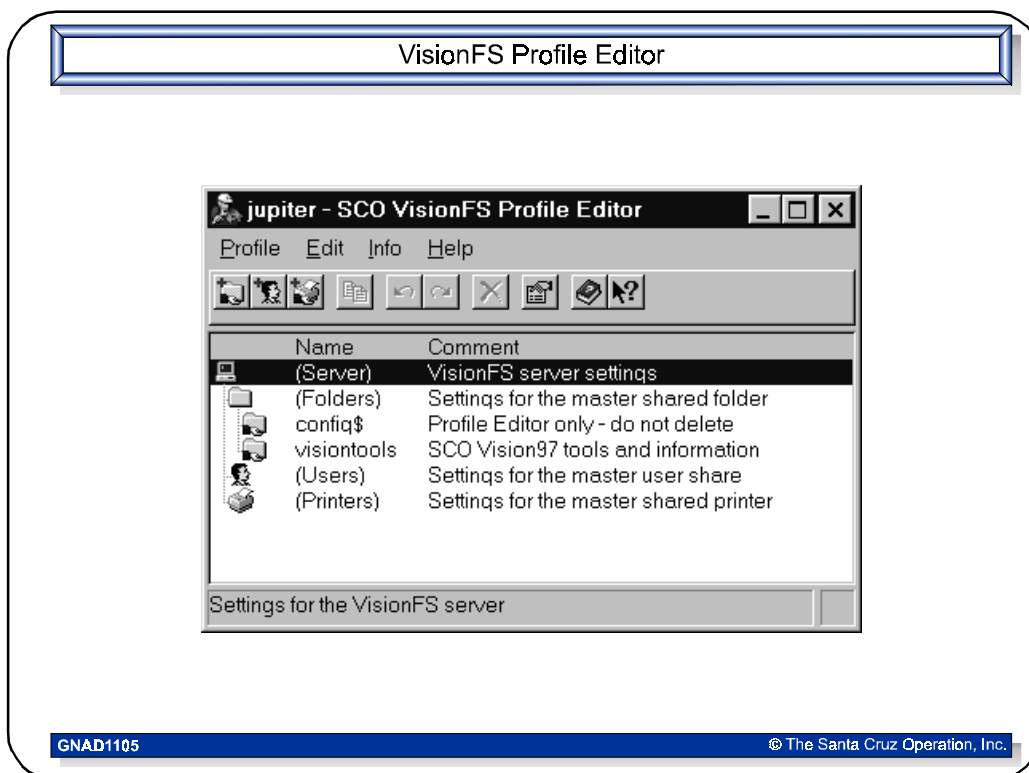
If you need to change the user name of the VisionFS administrator, or give the Windows user a different name from the UNIX user, enter this command on the UNIX system:

**/usr/vision/bin/visionfs setup**

This prompts you to answer several questions including what name you want to use for the VisionFS administrator.

By default, VisionFS creates a shared folder called *visiontools*. This share is used to provide access to the **Profile Editor**. You can run the **Profile Editor** directly from the VisionFS server without copying it to the Windows system.

AV 11-5



The **Profile Editor** is shared as *visiontools\visionfs\profedit.exe*. This file is located on the UNIX server in */usr/vision/bin/pctools/en\_US/visionfs*.

By default, the VisionFS server starts automatically when UNIX boots. If your VisionFS server is not running, you must start it before you can access the *visiontools* share. To start VisionFS, enter this UNIX command:

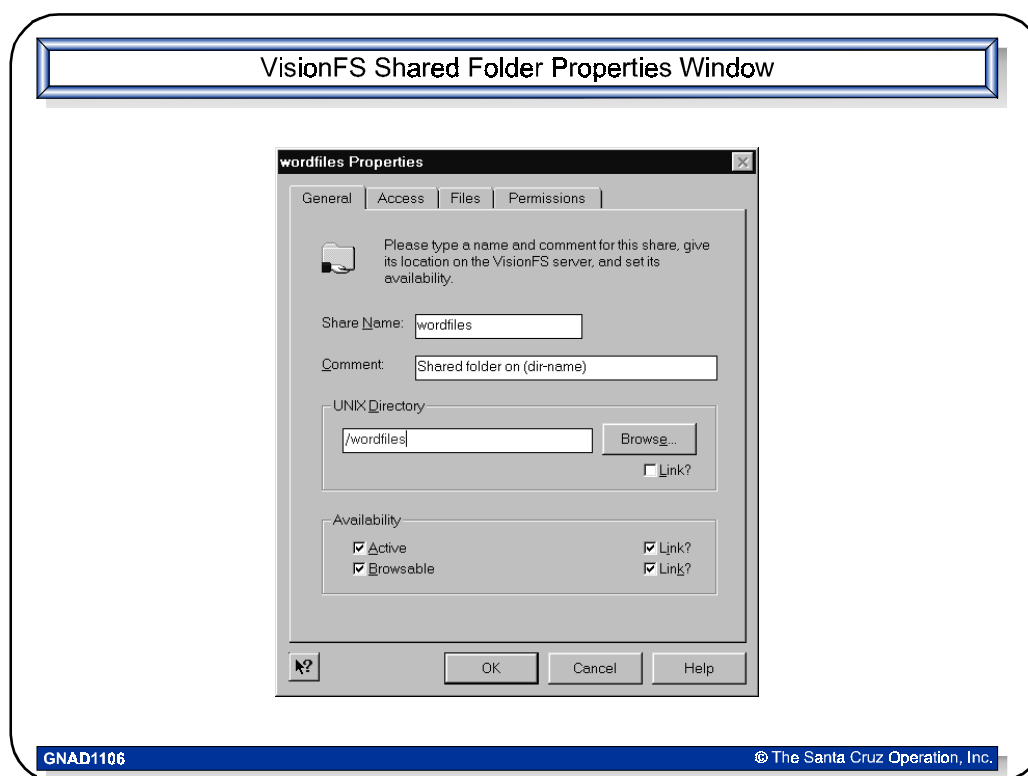
**/usr/vision/bin/visionfs start**

### **Exercise: Running the profile editor**

---

- |         |                                                                                                     |
|---------|-----------------------------------------------------------------------------------------------------|
| Step 1. | Log in to a Windows system as the VisionFS administrator. By default, the username is <b>root</b> . |
| Step 2. | Access the VisionFS server.                                                                         |
| Step 3. | Open the <i>visiontools</i> share.                                                                  |
| Step 4. | Open the <i>visionfs</i> folder.                                                                    |
| Step 5. | Run the VisionFS <b>Profile Editor</b> . Double-click on:<br><br><b>profedit.exe</b>                |

AV 11-6



## 11.2.7 Sharing files and directories with VisionFS

---

By default, VisionFS shares each user's home directory to that user. This is called an *automatic user share*. For example, when the user **alison** browses a newly installed VisionFS server from a client system, two shares are available:

- *visiontools*, which is `/usr/vision/bin/pctools/en_US` on the server
- *alison*, which is `/home/alison` (or **alison**'s home directory) on the server

To add a new shared directory, follow these steps:

- Step 1. Create a directory on the UNIX server. This must be done from the UNIX system.
- Step 2. Run the VisionFS **Profile Editor** for the correct server.
- Step 3. Select **Edit** ⇒ **New Shared Folder** from the menu.
- Step 4. Enter the name that users will see in the “Share Name” field.
- Step 5. Enter the UNIX pathname in the “UNIX Directory” field.
- Step 6. Click on **OK**.
- Step 7. Select **Profile** ⇒ **Update Server** from the menu.

Your changes will not be visible to clients until the server has been updated. The server can be updated even if it is in use by other users.

### Exercise: Sharing a directory

---

On your UnixWare 7 system, create a directory called `/wordfiles`. Use the **Profile Editor** to share this directory with the same share name.





### 11.2.7.1 Default settings and the master shared folder

---

By default, the settings for each share are dynamically linked to default settings. The default settings are available through an item called the *master shared folder*. You can control which settings are linked to the defaults through the **Link** buttons available in the **Profile Editor**:

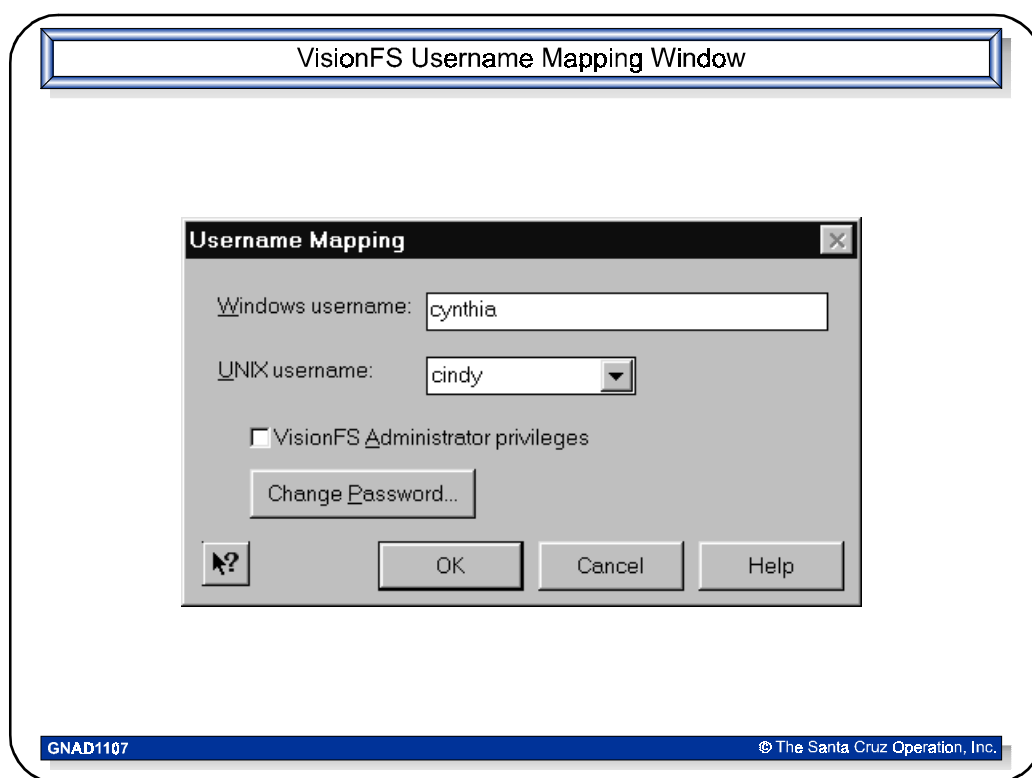
1. If the **Link** button is checked, the setting is automatically set to the default value stored in the master shared folder.
2. If the **Link** button is not checked, you can change its setting.

To change the master settings in the **Profile Editor**:

Step 1.           Select **Settings for the master shared folder** from the list.

Step 2.           Select **Edit** ⇨ **Properties** from the menu.

AV 11-7



### 11.2.7.2 Username mapping

---

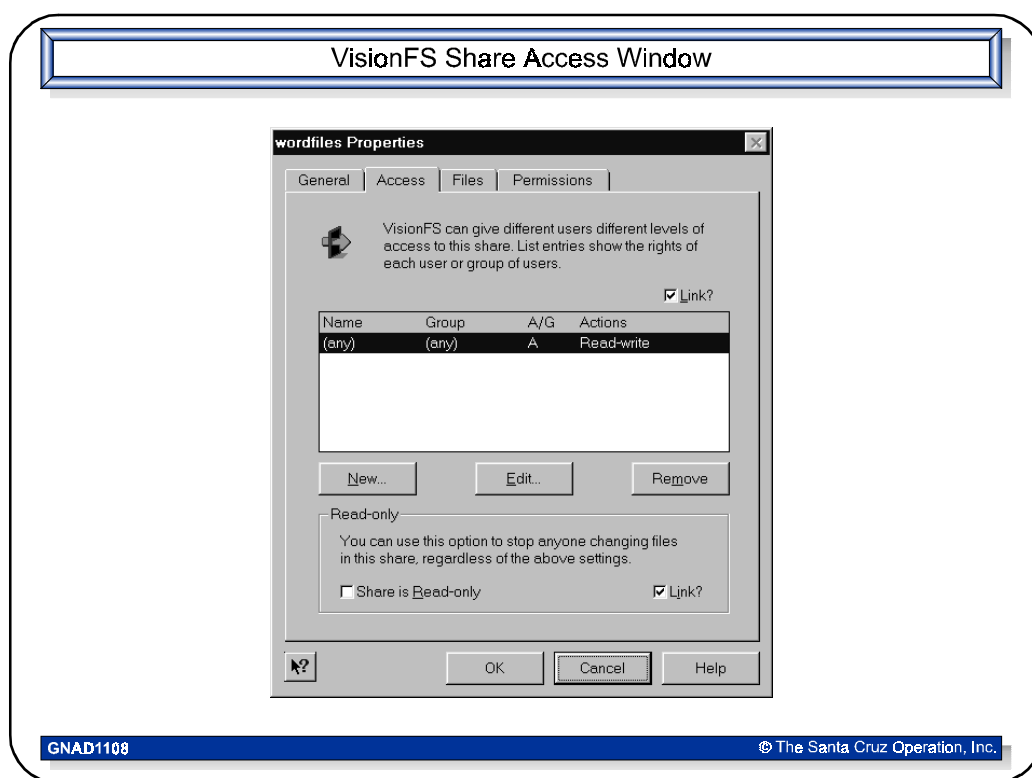
You might have users that have accounts on both your UNIX and Windows systems but their username can differ. To allow these users authenticated access to the VisionFS shares, you must create a *username mapping*.

For example, to map from the Windows user **cynthia** to the UNIX user **cindy**:

- Step 1.       Select **VisionFS server settings** from the **Profile Editor**.
- Step 2.       Select **Edit** ⇒ **Properties** from the menu.
- Step 3.       Select the **Users** tab.
- Step 4.       Click on **New**.
- Step 5.       Enter **cynthia** in the “Windows username” field.
- Step 6.       Enter **cindy** in the “UNIX username” field.
- Step 7.       Click on **OK**.

Note that you have the option of making this user a VisionFS administrator. Only do this if you would trust the user with the **root** password on your UNIX system because this user can create a read-write share of your root directory.

## AV 11-8



### 11.2.7.3 Limiting access to a shared folder

---

You might want to restrict which users have read or write access to a given folder. Windows users are divided into two categories:

*authenticated* Users who have a corresponding UNIX user account

*guest* Users who do not have a corresponding UNIX user account

To determine what access a user has to a share, you must consider:

- VisionFS share access is configured with the VisionFS **Profile Editor**.
- Normal UNIX permissions are configured with the **chmod(1)**, **chown(1)**, and **chgrp(1)** commands on the UNIX server.

For a type of access to be granted, both the VisionFS share access and the UNIX permissions must grant that type of access.

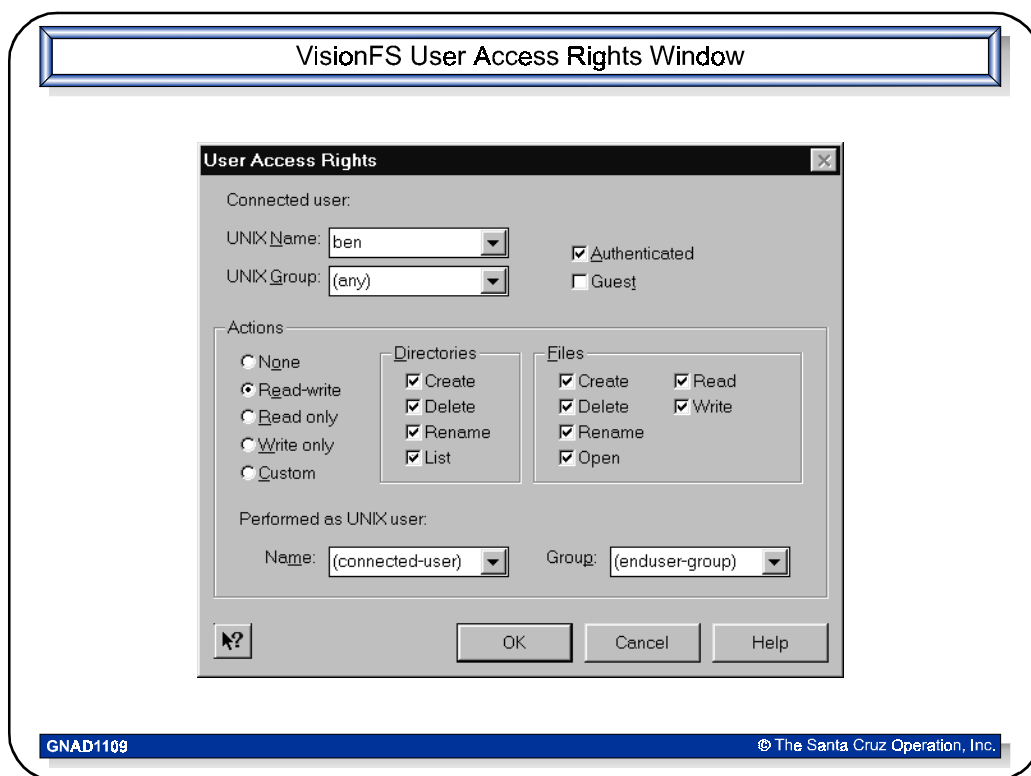
By default, guest users do not have access to VisionFS shares.

By default, authenticated users have read-write access to a shared folder through VisionFS. To actually read or write shared files, the UNIX permissions must also grant that user the appropriate rights.

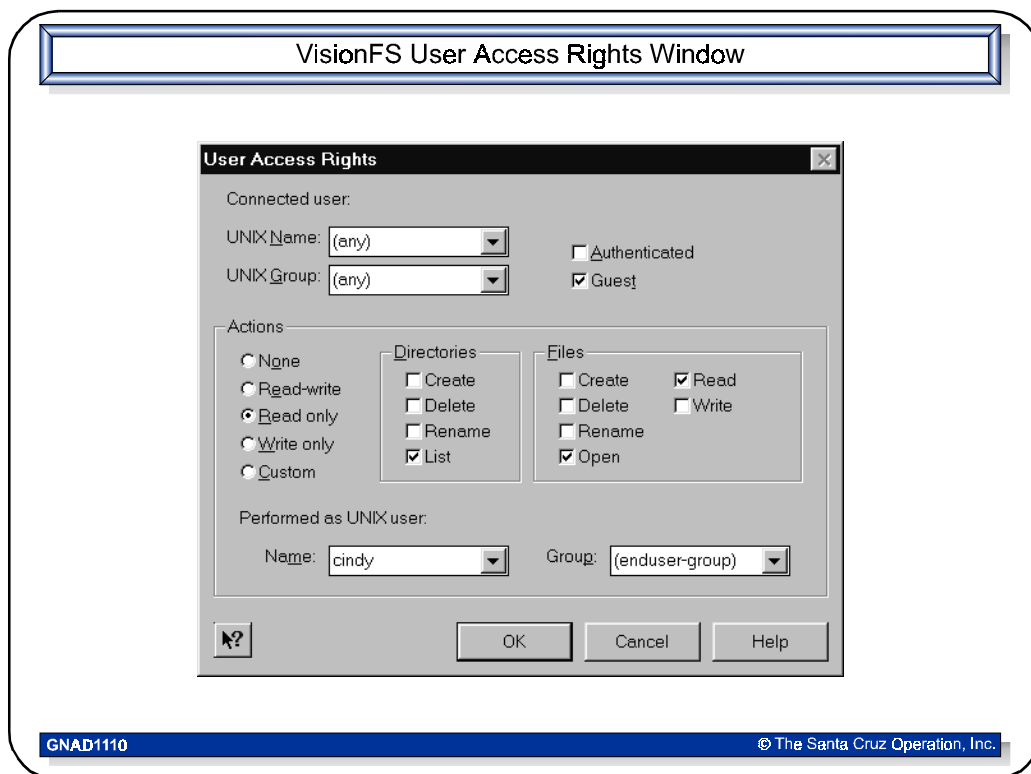
To configure VisionFS shared folder access:

- Step 1. Run the VisionFS **Profile Editor**.
- Step 2. Select the share you want to configure.
- Step 3. Select **Edit** ⇒ **Properties** from the menu.
- Step 4. Select the **Access** tab from the **Properties** window. This displays the window shown AV.

AV 11-9



AV 11-10



For example, to change the default access to read-only for a share:

- Step 1. Select the line corresponding to the “Name” of **(any)**.
- Step 2. Click on **Edit**.
- Step 3. Select **Read only** under the “Actions” heading.
- Step 4. Click on **OK**.

To add read-write access for a particular user named **ben**, as shown in AV 11-9:

- Step 1. Click on **New**.
- Step 2. Select the user **ben** from the “UNIX Name” field.
- Step 3. Select **Read-write** under the “Actions” heading.
- Step 4. Click on **OK**.
- Step 5. Set permissions on the UNIX system to grant **ben** read-write access.

To grant read-only access to guest users as shown in AV 11-10:

- Step 1. Click on **New**.
- Step 2. Deselect the **Authenticated** checkbox. The **Guest** checkbox is selected for you.
- Step 3. Select **Read only** under the “Actions” heading.
- Step 4. Select the UNIX user whose permissions will be used for the guest users under the “Performed as UNIX user” heading.
- Step 5. Click on **OK**.

### 11.2.8 Sharing printers using VisionFS

---

VisionFS can be used to share printers attached to the both the UNIX server system and Windows client machines.

This means that the Windows client users can access printers on the UNIX system, and UNIX users can print to printers attached to Windows machines.

### 11.2.8.1 Sharing server printers using VisionFS

---

When the VisionFS server starts, it automatically shares all of the printers found on the UNIX server. Each printer on the UNIX server has an icon displayed on the Windows client. To display shared printers, access your VisionFS server by double-clicking on the **Network Neighborhood** icon.

If your VisionFS server is listed, double-click on its icon.

If your VisionFS server is not listed, recall the alternate method for accessing the server:

Step 1.        Select **Start** ⇒ **Run** from the Windows 95 menu.

Step 2.        Enter:

`\\servername`

where *servername* is the name of your VisionFS server.

Note that the VisionFS server normally starts when the UNIX system boots. If you have added new printers since the last reboot, restart VisionFS so that it detects the new printers.



To restart the VisionFS server, select **Profile** ⇒ **Restart Server** from the **Profile Editor** menu on your Windows client. To restart VisionFS from the UNIX command line, enter:

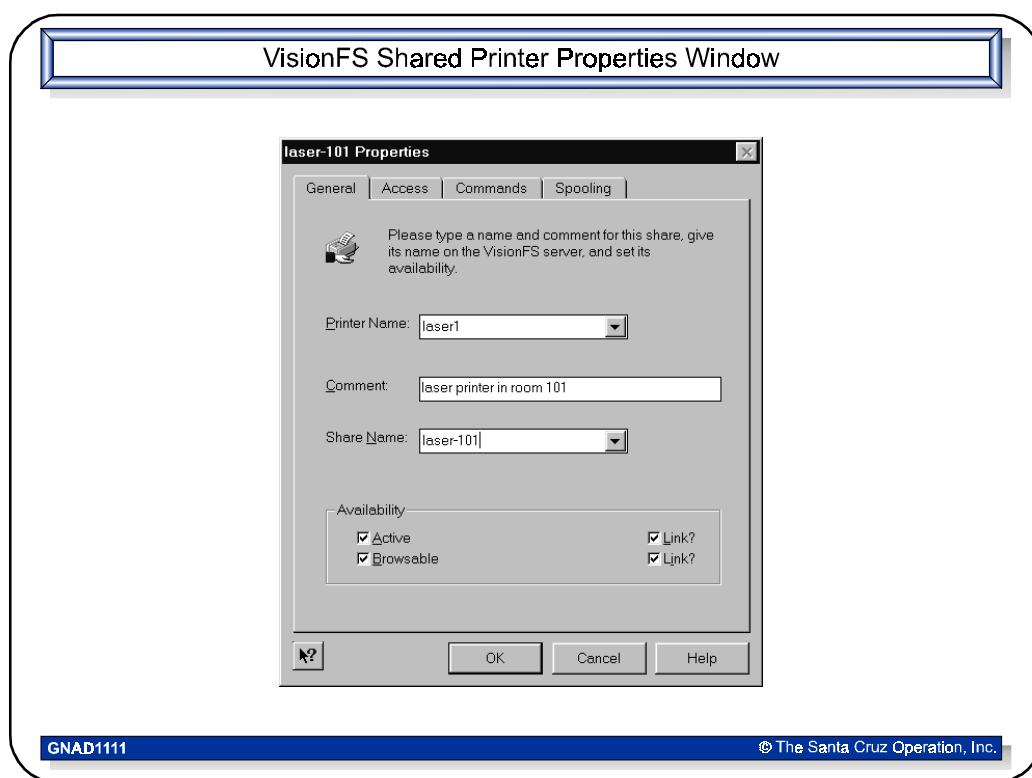
**/usr/vision/bin/visionfs stop**

**/usr/vision/bin/visionfs start**

To use a shared printer from Windows 95, you must add a network printer to your Windows system. To start the setup process, double-click on the printer icon and follow the normal Windows prompts.

Network printer setup can also be started from Windows using **Start** ⇒ **Settings** ⇒ **Printers** ⇒ **Add Printer**.

AV 11-11



By default, newly-shared printers use the settings for the master shared printer. To change the master settings in the **Profile Editor**:

Step 1.       Select **Settings for the master shared printer**.

Step 2.       Select **Edit** ⇒ **Properties**.

To customize settings for a particular printer, create an explicit share rather than using the automatic one. To explicitly share a printer:

Step 1.       Select **Edit** ⇒ **New Shared Printer**.

Step 2.       Select the UNIX name for the printer in the “Printer Name” field.

Step 3.       Enter the name of the printer in the “Share Name” field if you use a different Windows name for the printer.

Step 4.       Click on **OK**.

To change the settings for an explicitly shared printer:

Step 1.       Select the shared printer to configure.

Step 2.       Select **Edit** ⇒ **Properties** from the menu.

### 11.2.8.2 Sharing client printers using VisionFS

---

You can have printers attached to your Windows client systems and make them available to UNIX applications. VisionFS can provide UNIX access to client printers. The client printers are shared using the built-in printer sharing tools found in Windows 95 and other versions of Windows. VisionFS is used to provide access to the client printers from the UNIX command line.

Configuring client printer sharing involves three steps:

1. Configure Windows to share printers.
2. Configure Windows to share a particular printer.
3. Use VisionFS to access the shared client printer.

To share printers from Windows 95:

- Step 1. Right-click on the **Network Neighborhood** icon.
- Step 2. Select **Properties** from the pop-up menu.
- Step 3. Click on **File and Print Sharing**.
- Step 4. Select "I want to be able to allow others to print to my printer(s)."
- Step 5. Click on **OK**.
- Step 6. Select the **Access Control** tab.
- Step 7. Click on **Share-level access control**.
- Step 8. Click on **OK**.
- Step 9. Reboot if needed.

To share a particular printer with Windows 95:

- Step 1.       Select **Start** ⇒ **Settings** ⇒ **Printers**.
- Step 2.       Right-click on the printer you want to share.
- Step 3.       Select **Sharing** from the pop-up menu.
- Step 4.       Click on **Shared As**.
- Step 5.       Supply a name for this printer in the “Share Name” field.
- Step 6.       Leave the “Password” field blank.
- Step 7.       Click on **OK**.

To print to the shared printer from the UNIX command line, use the command:

```
visionfs print //client/printer filename --user guest --password
```

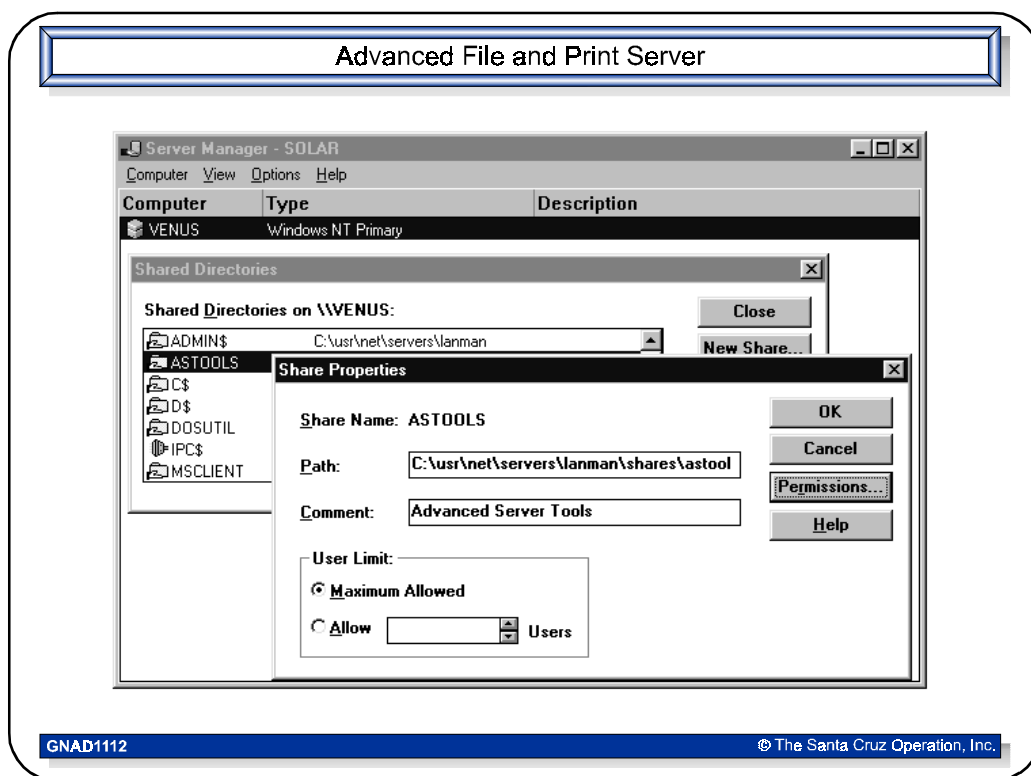
where:

- client*       Is the name of the client system that is sharing the printer
- printer*      Is the share name for the printer
- filename*     Is the UNIX file that you want to print

The **visionfs** command is found in the */usr/vision/bin* directory.

For more convenient use, you can write a shell script that runs the above command. Alternatively, you can write a printer interface script so the shared printer can be accessed using the **lp(C)** command.

AV 11-12



## **11.2.9 Advanced File and Print Server**

---

VisionFS is not the only CIFS server package available for UnixWare 7. However, it is the only one included in the base UnixWare 7 distribution.

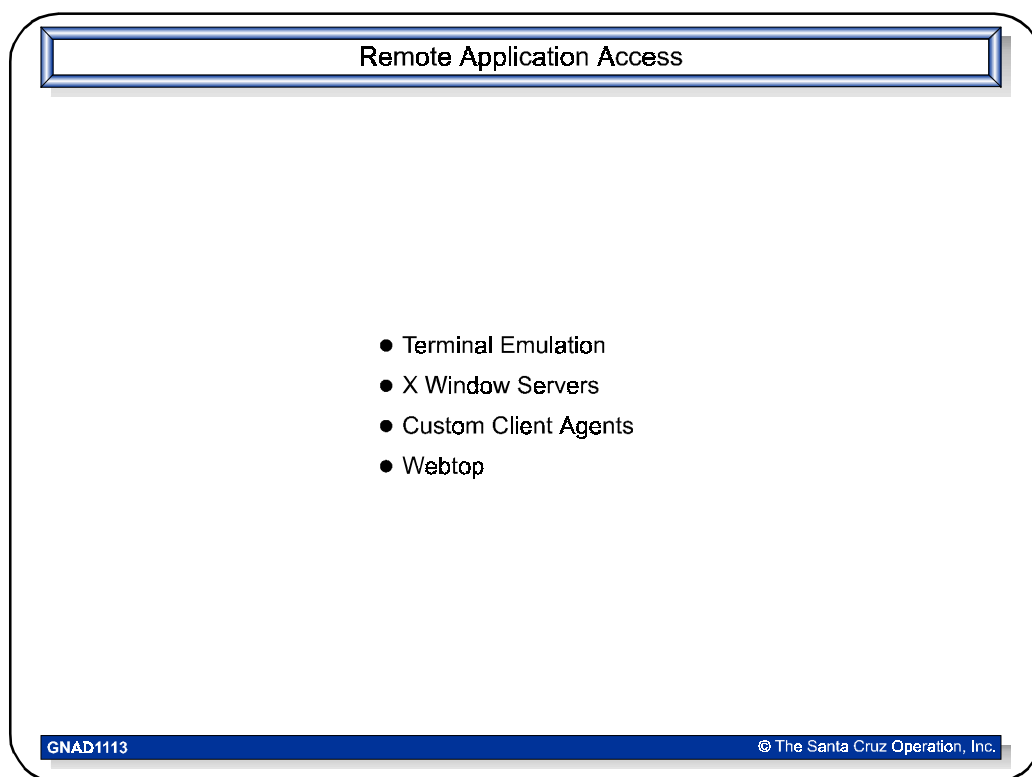
An alternative available is SCO Advanced File and Print Server (AFPS). AFPS is a more sophisticated product than VisionFS. However, the interface seen by your Windows users is the same because both use the native CIFS based tools included with Windows. Advantages of using AFPS over VisionFS include:

- AFPS provides Windows NT style user login authentication. An AFPS system can serve as the Primary or Backup Domain Controller for your Windows Network. VisionFS does not provide Windows login authentication.
- AFPS uses the Windows NT concept of a domain. A domain controlled by your AFPS server can trust, or be trusted by, a domain controlled by another AFPS server or a Windows NT server.
- AFPS supports NetBEUI as a transport protocol in addition to NetBIOS over TCP/IP. VisionFS only supports NetBIOS over TCP/IP. NetBEUI can be more appropriate on some networks, particularly if you are already using it for other services and do not want to use client resources to run a TCP/IP stack.
- Two AFPS systems can be configured to replicate data between them. This allows you to load-balance your network traffic between multiple servers without manually copying files and keeping them up to date.
- The AFPS administration tools are identical to the Windows NT administration tools. If you are already familiar with Windows NT administration, very little training is required.

Disadvantages of AFPS relative to VisionFS include:

- AFPS is not included with UnixWare 7 and must be purchased separately.
- AFPS is more complicated to administer than is VisionFS.

AV 11-13





## **11.3 Remote Application Access**

---

In addition to (or instead of) sharing files and printers, you might also want to provide remote access to applications that run on your UNIX system.

Reasons to run applications on a UNIX server include:

- Your server has the required CPU power, memory capacity, disk space, or other hardware resources that are not available on your client machines.
- You have applications that are available for UNIX but are not available for the client operating system, or they are more reliable under UNIX than the client operating system.
- You might need to provide access to an application to many people at once.

A variety of strategies for providing client users with access to UNIX applications include:

- Terminal Emulation
- X Window Servers
- Custom Client Agents
- Webtop

### 11.3.1 Terminal emulation

---

If your application uses a text-only interface, one approach is to use terminal emulation software on the Windows client. The terminal emulation software gives your Windows client the same capabilities as a dumb terminal, such as a VT100 or Wyse 60.

Most terminal emulation programs for Windows display one or more windows, each providing a separate login shell on the UNIX server. In each window, the user can invoke text-based applications that run on the UNIX server.

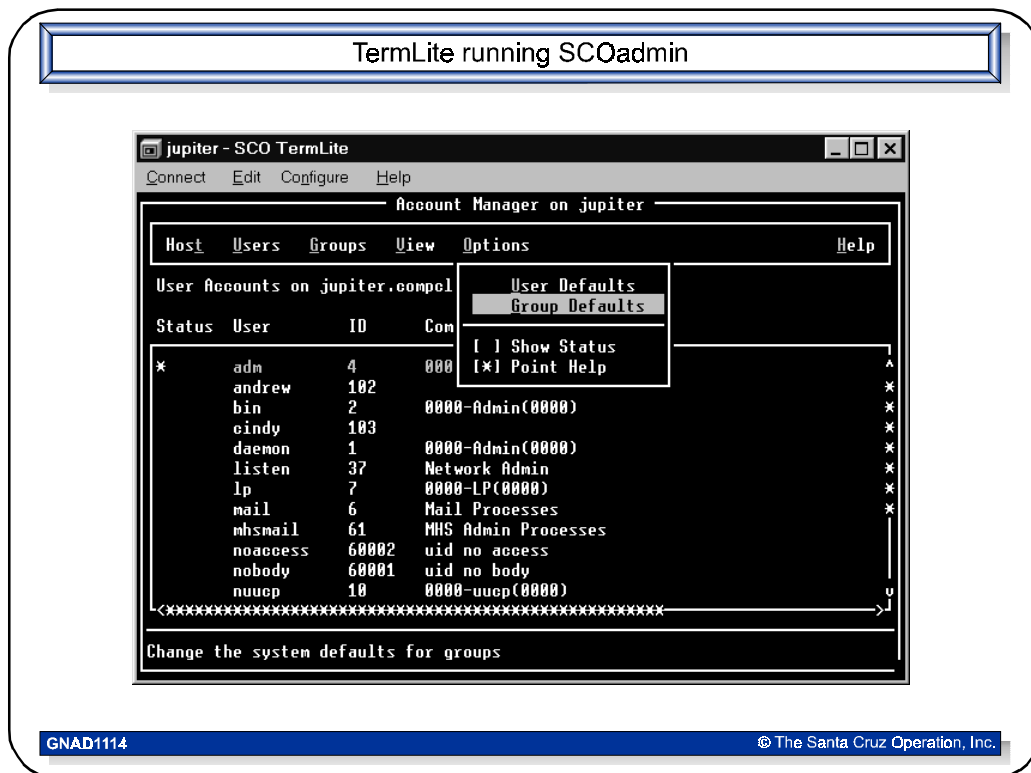
Advantages of terminal emulation include:

- It can be very easy to configure.
- It is usually faster and produces less network traffic than X Window servers, as it often takes less information to describe a text screen than a graphic one.
- It provides access to any text-based application on the server; you do not need a separate client agent for every application. This simplifies installation and support over a custom client approach.
- Your users can use a wide variety of client platforms, including DOS and dumb terminals, in addition to Windows machines. Nearly every operating system has terminal emulation support of some kind.

Disadvantages of terminal emulation include:

- Only plain-text applications are supported. Many users prefer a graphical interface.
- It usually produces more network traffic than custom client agents because every keystroke the user types must be sent to the server (often as a separate network packet) and every character of every screen must be sent back to the user from the server.

AV 11-14



## TermLite

---

TermLite is a lightweight terminal emulator for Windows that is included with UnixWare 7. It emulates the ANSI, VT100, and VT320 terminal types, among others.

A version of TermLite is installed on your UnixWare 7 system when VisionFS is installed, and made available in a shared folder. It can also be installed separately from the Optional Services CD-ROM.

To run TermLite from Windows 95, you have two options:

- Run TermLite directly from the share, leaving the executable file on the server.
- For VisionFS, select **Network Neighborhood** ⇒ *server* ⇒ **visiontools** ⇒ **termlite** ⇒ **termlite.exe**

This can also be done using the MS-DOS command line (or **Start** ⇒ **Run**). To do this, enter:

**\\server\visiontools\termlite\termlite.exe**

- Run the installation program from the share, which installs the TermLite executable on your client machine.
- For VisionFS, select **Network Neighborhood** ⇒ **server** ⇒ **visiontools** ⇒ **termLite** ⇒ **setup.exe**

This can also be done using the MS-DOS command line (or **Start** ⇒ **Run**). To do this, enter:

**\\server\visiontools\termLite\termLite.exe**

While TermLite is running, to open a connection to a remote system:

- Step 1. Select **Connect** ⇒ **Connect to** from the menu.
- Step 2. Enter the name of the UNIX host in the “Host” field.
- Step 3. Click on **Connect**.
- Step 4. Log in to the UNIX system when prompted to do so.

To change the terminal type that TermLite emulates:

- Step 1. Select **Configure** ⇒ **Terminal** from the TermLite menu.
- Step 2. Click on **Identity**.
- Step 3. Select a terminal type from the “Identity Type” list.

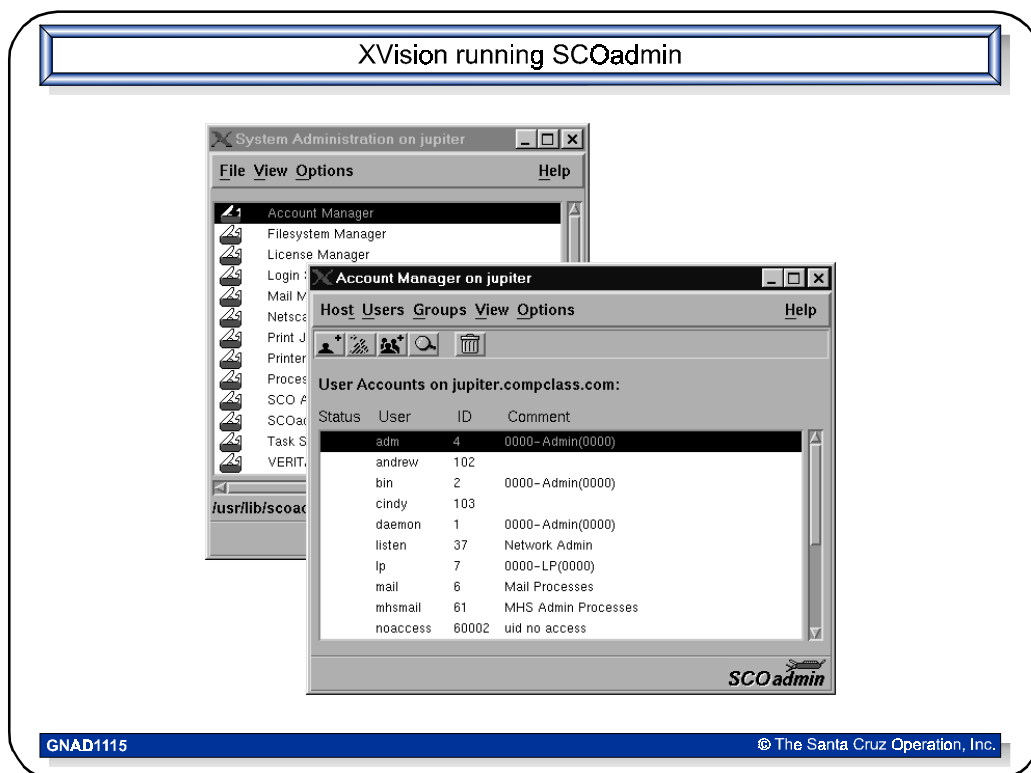
Much more sophisticated terminal emulation software is available from a variety of sources. One product is SCO TermVision™.

**Exercise: Running TermLite from the VisionFS server**

---

- Step 1. Run TermLite from a Windows system leaving the TermLite executable on the VisionFS server.
- Step 2. Run programs such as **vi(1)**, **scoadmin(1M)**, and **setcolor(1)**. Adjust the terminal type if needed.
- Step 3. Select **Configure** ⇨ **Terminal** from the menu.
- Choose yellow text on a blue background.
  - Change the number of lines displayed to 35.
- Step 4. Resize the window so the font is a readable size.
- Step 5. Experiment with other TermLite configuration options.

AV 11-15





### 11.3.2 X Window servers

---

One feature that terminal emulators cannot provide is a graphical user interface. UNIX graphical applications are usually written to use the X Window system to display their graphical interfaces.

The X Window system consists of a client/server pair, where:

- The *client* is the application that requires access to a graphic display. Usually, this will be an application running on the UNIX machine.
- The *server* is software that draws on a graphic display. The X server software must run on the machine that provides the graphic display. In our examples, the X server runs on a Microsoft Windows machine.

The X Window system was designed to allow the client application and the display server to run on different machines connected by a network. To use Microsoft Windows as the display platform, you need software that will accept network requests to display graphics. This software is called an *X Window server*.

X Window server software is included with UnixWare 7. One example of X Window server software for Microsoft Windows is SCO XVision. It is not included with UnixWare 7.

Advantages of using X to provide remote application access include:

- Users see fully graphical applications on their screens. Microsoft Windows users are often most comfortable with graphical interfaces, and might not be satisfied with the text-only interface provided by a terminal emulator.
- X Window servers are available for a variety of graphical operating systems, not just Microsoft Windows and UNIX. Users can even use X-terminals, network devices which are just X Window servers with graphic displays.
- An X Window server can be used to display any X Window applications running on your UNIX system. You do not need to install separate Microsoft Windows agents for every application.

Disadvantages of the X Window system include:

- X Window server software is not included with Microsoft Windows, Windows 95, or Windows NT, so additional software must be obtained.
- X imposes a relatively heavy load on your network. Because graphical interfaces are often complex, and require many individual drawing commands to render, many packets must cross the network to draw each screen.

In addition, every keystroke, mouse movement, and mouse click might be sent back to the application for possible processing. For example, moving the mouse in and out of the application's window without clicking will normally send a steady stream of packets back to the application.

### 11.3.3 Custom client agents

---

Another option for providing remote access is to use a custom client agent to interface with a server application. Features of this arrangement include:

- The agent runs natively on the client platform, such as Windows.
- The agent receives user input and displays program output.
- The agent uses the network to communicate with the server-side application.

Examples of custom client agents include:

- The VisionFS **Profile Editor**, a native Windows executable, that communicates over the network with the VisionFS server and allows you to configure the server-side application.
- A database query agent, where:
  - The user forms a database query with a Windows interface.
  - The client agent sends the query to a database on the UNIX server.
  - The UNIX database application processes the query.
  - The result of the query is sent back to the client for display.

Note that the only network traffic is the query and result data itself; all formatting is added on the client side and network traffic is not generated while the user is constructing the query or reading the result.

- A mathematics environment (such as Mathematica) where:
  - The user uses a client agent to construct an equation.
  - The client transfers the completed equations to a server-side engine.
  - The server-side engine computes solutions (this can be CPU intensive).
  - The solutions are sent back to the client for display.

Advantages of custom client agents include:

- Network traffic is minimized because the custom client can handle the user interface and display formatting without bothering the server or the network.
- Because the client agent is written for a specific client platform, it can best take advantage of the user interface used on the client system. In contrast, an X Window application can appear strange to a Microsoft Windows user.
- The user often does not need to know that a server is involved in their work, which reduces user complexity and training costs.

Disadvantages of custom client agents include:

- The client must be written for a particular operating system, such as Microsoft Windows. Supporting multiple client platforms means porting the client agent. In some cases, this can be avoided by writing the client in a platform-independent environment such as Java.
- A separate client agent is required for every application, which increases the complexity of installation and support.

### 11.3.4 Webtop

---

The UnixWare 7 Webtop is a new, innovative web-based interface that allows a UnixWare 7 system and its applications to be accessed securely from a Java™ enabled browser. Through the Webtop, clients as diverse as Network Computers and PCs are able to tap in to the full power of UnixWare 7.

The Webtop gives end users quick and easy access to UnixWare 7 applications, without requiring any UnixWare 7 experience. In addition, the Webtop improves user efficiency because applications are maintained by a central administrator, thus freeing users from needing the skills or time to do so themselves. Designated "Webtop administrators" control which applications are available to users, making it possible for each user to receive a set of applications customized for his or her needs.

The Webtop also benefits system administrators, who can perform a wide range of system administration tasks without being tied to the console. The new System Troubleshooting interface provides an immediate overview of potential system difficulties and allows administrators to quickly resolve problems.

#### **Exercise: Webtop Demonstration**

---

---

Use a Netscape® browser to run through a Webtop demonstration at:  
<http://www.sco.com/unixware/webtop/demo/demo.html>.

## 11.4 Remote Boot Configuration

---

If you have many clients, or clients that move around the network frequently, it can be tedious to configure and reconfigure networking parameters on every system. UnixWare 7 can help by providing remote boot configuration services to your Windows (and other operating system) clients.

This section describes the use of the Dynamic Host Configuration Protocol (DHCP). A DHCP server answers broadcasts from a booting client with configuration parameters for the client to use.

Examples of the parameters DHCP can provide include IP address, netmask, broadcast address, domain name, DNS server, and default router.

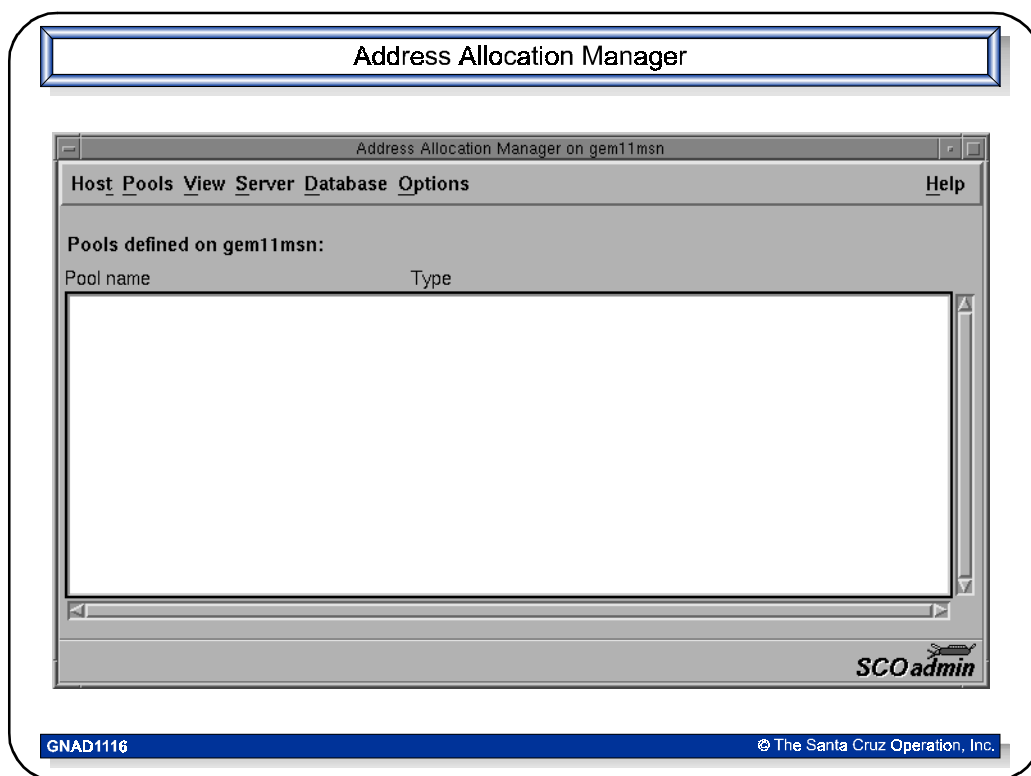
Benefits of using a remote boot configuration protocol, such as DHCP, include:

- You can avoid manual configuration of every client. This can save a considerable amount of administrator time.
- If a client is mobile and often connected to different networks, it can automatically configure itself for the local network without user intervention.
- If a global network parameter such as network address or domain name must be changed, it can be done easily from the server.
- Network conflicts, such as two clients attempting to use the same IP address, are much less likely because the server will not assign the same address to different clients at the same time.
- Temporary visitors with their own computers, such as students in a training classroom, can have their systems automatically configured for the local network.

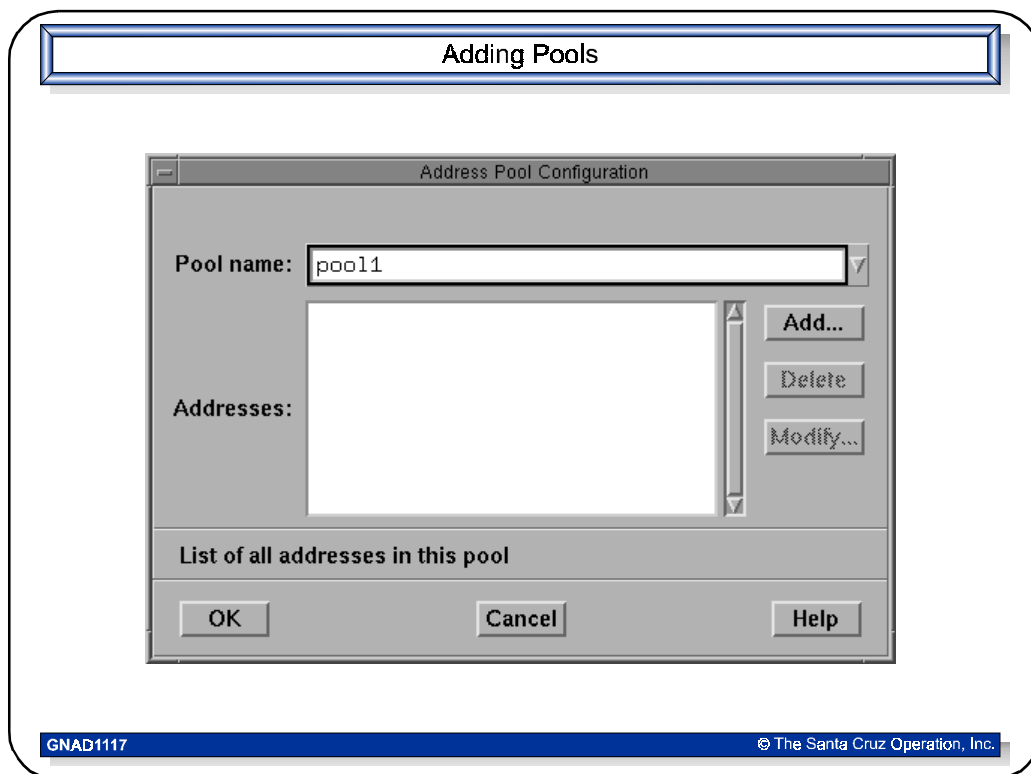
When a DHCP client boots, the following sequence of steps occurs:

1. The client broadcasts a request for its configuration parameters.
2. A DHCP server receives the broadcast and selects appropriate parameters.
3. The DHCP server requests an IP address from an Address Allocation Server (AAS).
4. The AAS responds with a currently unused IP address from its available pool.
5. The DHCP server responds to the client with the list of configuration parameters including the IP address to use.

AV 11-16



AV 11-17



### 11.4.1 Configuring an Address Allocation Server

---

An Address Allocation Server (AAS) is needed if a DHCP server is going to assign IP addresses from a pool. The AAS maintains a list of address pools, each of which contains one or more IP addresses.

The AAS can be used to assign addresses for both DHCP and PPP. However, a single pool cannot be used for both DHCP and PPP. This means that an IP address cannot float between a DHCP client and a PPP client. For more information on PPP, refer to Section 6.2.2, “Point-to-Point Protocol (PPP),” on page 6-21.

The AAS is configured using the **scoadmin(1M) Address Allocation Manager**.

To start the **Address Allocation Manager**:

- Step 1. Run **SCOadmin**.
- Step 2. Select **Networking** ⇒ **Address Allocation Manager**. This displays the window shown in AV 11-16.

#### 11.4.1.1 Creating address pools

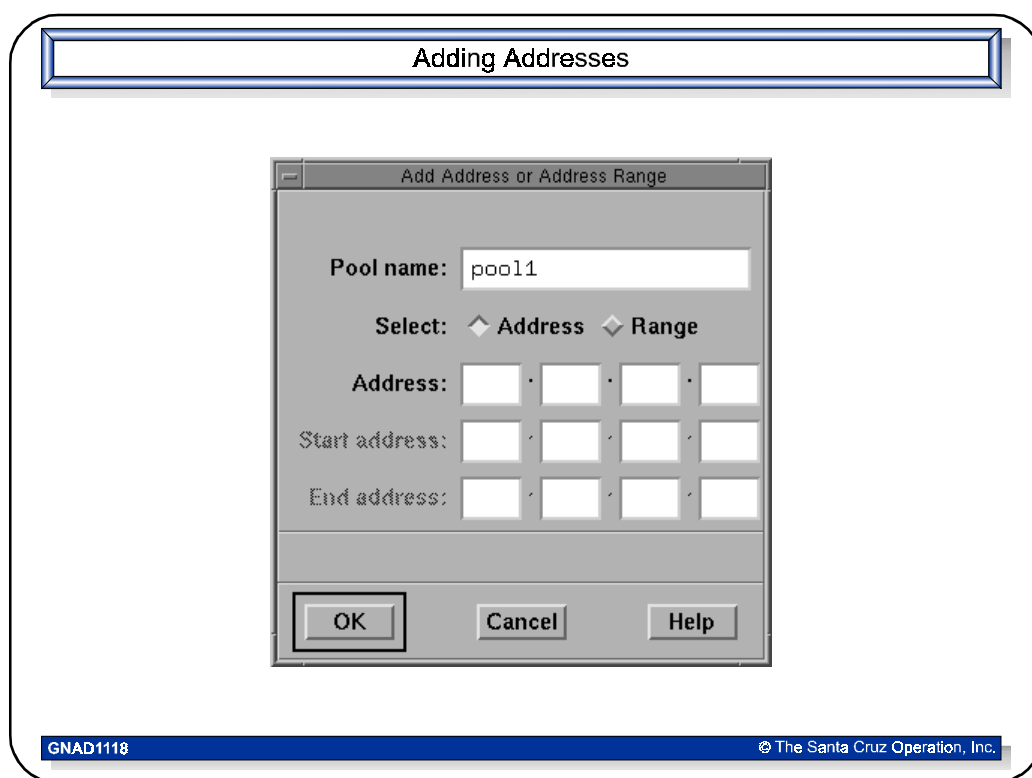
---

An *address pool* is a list of IP addresses. When a booting client needs to be assigned an IP address, the AAS selects an unused address from a pool and issues it to the client. You can create different pools, but the pools cannot overlap. Each pool is a collection of individual addresses and address ranges.

To create an address pool from the **Address Allocation Manager**:

- Step 1. Select **View** ⇒ **Pool Definition**.
- Step 2. Select **Pools** ⇒ **Add** to display the window shown in AV 11-17.
- Step 3. Supply a name for the pool in the “Pool name” field.
- Step 4. Click on **Add** to add addresses to the new pool.

AV 11-18





To add a single address:

- Step 1. Click on **Address**.
- Step 2. Enter the IP address in the “Address” field.
- Step 3. Click on **OK**.

To add a range of addresses:

- Step 1. Click on **Range**.
- Step 2. Enter the first address in the range in the “Start address” field.
- Step 3. Enter the last address in the range in the “End address” field.
- Step 4. Click on **OK**.

When you are finished, click on **OK** in the Address Pool Configuration window.

### **Exercise: Configuring an address pool**

---

Create a new address pool, using instructor-supplied IP addresses.

### 11.4.1.2 Starting the AAS daemon

---

The AAS daemon is **/usr/sbin/in.aasd**. By default, it is started at boot time if the configuration **/etc/inet/aasd.conf** file exists. The **/etc/inet/aasd.conf** file is created when you add the first pool with the **Address Allocation Manager**. However, the daemon is not started until you reboot or start the daemon explicitly.

To start the AAS daemon:

Step 1. Start the **Address Allocation Manager** from SCOadmin.

Step 2. Select **Server** ⇒ **Start** from the menu.

Note that you cannot start the AAS daemon until after you have configured at least one address pool.

#### **Exercise: Starting the AAS daemon**

---

Start the AAS daemon on your system.

## 11.4.2 Configuring a DHCP server

---

DHCP does much more than assign an IP address to each client. This section discusses in detail the features of DHCP and how to configure a server using the **DHCP Server Manager**.

### 11.4.2.1 DHCP leases

---

A DHCP server leases an IP address to a client. The lease may be for a limited or unlimited period of time. If a lease is for a limited period of time, it goes through three phases:

|         |                                                                                                                                                                                |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BOUND   | The client is using the leased address.                                                                                                                                        |
| RENEWAL | The limited-term lease is about to expire. The client asks the server to renew the lease and the server might, or might not acquiesce.                                         |
| REBIND  | The original server has not renewed the lease. The client broadcasts again for a DHCP server, which can be answered by either the original server, or a different DHCP server. |

A leased IP address becomes available in these cases:

- A client shuts down normally.
- A client releases an address voluntarily.
- A server detects that the client is down.
- A server does not renew an expiring lease.



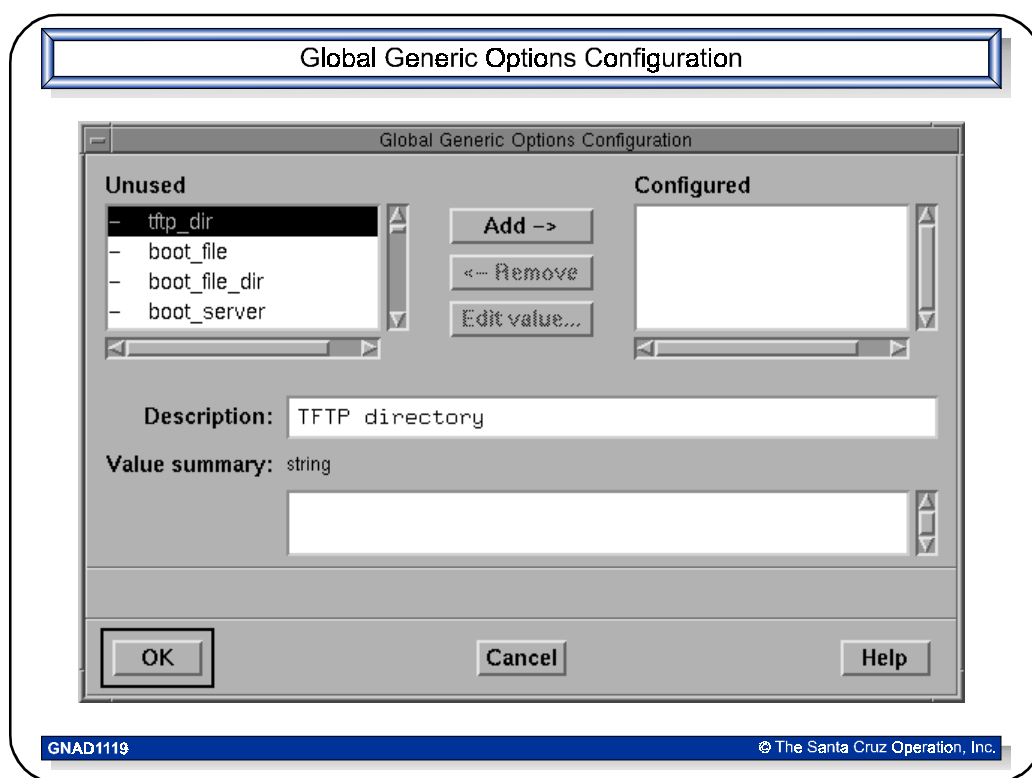
### **11.4.2.2 The DHCP class hierarchy**

---

DHCP uses a hierarchical model to assign parameters to clients. Three levels of the hierarchy are:

- Global
  - Applies to all clients
  - Can be overridden by a setting at a lower level (subnet or client)
  - Can be used for such options as domain name, DNS server, and default router
- Subnet
  - Applies to all clients in a network or subnet
  - Can be overridden by a client specific setting
  - Can be used to specify a pool of available IP addresses for the subnet
- Client
  - Applies to a specific client
  - Is identified by a MAC address, such as an Ethernet address
  - Can be used to assign a static IP address to a particular client system

AV 11-19



### 11.4.2.3 Accessing the DHCP Server Manager

---

DHCP is configured through the **scoadmin(1M) DHCP Server Manager**. To start the **DHCP Server Manager**:

- Step 1.       Run **scoadmin(1M)**.
- Step 2.       Select **Networking** ⇒ **DHCP Server Manager**.

### 11.4.2.4 DHCP global options

---

Global options can affect all client systems. A full list of the many global options and their meanings can be found in RFC 1533. This course only discusses options used by Windows 95 systems.

To set a value for a DHCP global option using the **DHCP Server Manager**:

- Step 1.       Select **View** ⇒ **Global Options**.
- Step 2.       Select **Entry** ⇒ **Add**. This displays the window shown in AV 11-19.
- Step 3.       Select an option from the “Unused” list.
- Step 4.       Click on **Add** to display the **Edit Option** window.
- Step 5.       Supply a value for the option.
- Step 6.       Click on **OK**.
- Step 7.       Configure additional options if desired. Click on **OK** in the **Global Generic Options Configuration** window.

The DHCP global options that are understood by Windows 95 include:

|    |                 |                                             |
|----|-----------------|---------------------------------------------|
| 3  | routers         | IP addresses of default gateways or routers |
| 6  | dns_servers     | IP addresses of DNS name servers            |
| 15 | domain          | DNS domain name                             |
| 28 | broadcast_addr  | IP broadcast address                        |
| 44 | nb_name_servers | IP address of NetBIOS name servers          |
| 46 | nb_node_type    | NetBIOS node type                           |
| 47 | nb_scope        | NetBIOS scope identifier                    |



In the **DHCP Server Manager**, the options are listed in order by code number. The code numbers are provided in the above list to help you locate these options.

You cannot select a global IP address pool from which to assign client IP addresses. IP addresses can be specified at the subnet or client levels.

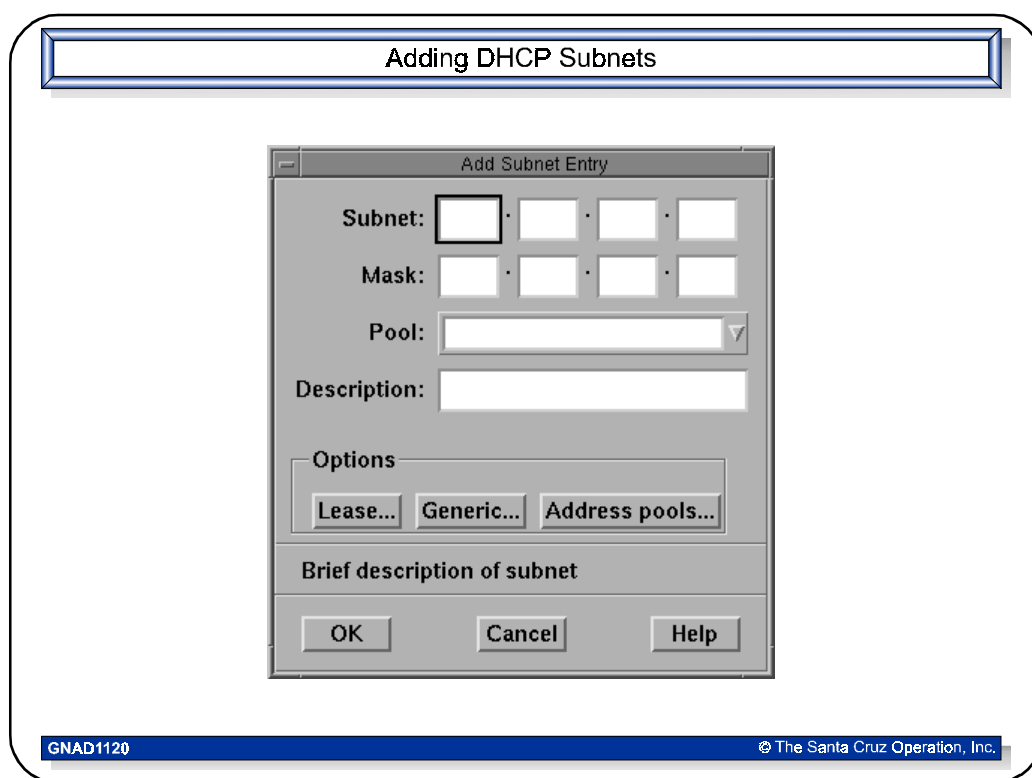
### **Exercise: Configuring DHCP global options**

---

Use the **DHCP Server Manager** to set the following options to instructor-assigned values:

- 3 routers
- 6 dns\_servers
- 15 domain
- 28 broadcast\_addr

AV 11-20



### 11.4.2.5 DHCP subnet options

---

Most DHCP clients on an IP network or subnet share certain DHCP parameters. To set DHCP parameters for a subnet with the **DHCP Server Manager**:

- Step 1.        Select **View** ⇒ **Subnets**.
- Step 2.        Select **Entry** ⇒ **Add**. This displays the window shown in AV 11-20.
- Step 3.        Enter the IP address of the subnet in the “Subnet” field. The host portion of the IP address must contain zeroes. For example, given:
- A class B network with address of 130.100
  - A netmask of 255.255.255.0
  - The “Subnet” field should contain **130 . 100 . 1 . 0** for subnet 1.
- Step 4.        Enter the netmask of the subnet in the “Mask” field.
- Step 5.        Select the name of the address pool to use in the “Pool” field.
- If you have not already created an address pool, you can create one from this form by clicking on **Address pools** in the **Options** area.
- Step 6.        Enter a description of the subnet in the “Description” field. This field is optional.
- Step 7.        Modify IP address lease parameters for all machines on this subnet by clicking on **Lease** in the **Options** area. See Section 11.4.2.7, “Address lease parameters,” on page 11-73 for more information on these parameters.
- Step 8.        Configure the DHCP options common to all machines on this subnet by clicking on **Generic** in the **Options** area. Use this to override global options for this subnet. For example, you could have a different default router for each subnet. See Section 11.4.2.4, “DHCP global options,” on page 11-63 for more information about these options.
- Step 9.        Click on **OK** to complete the configuration.

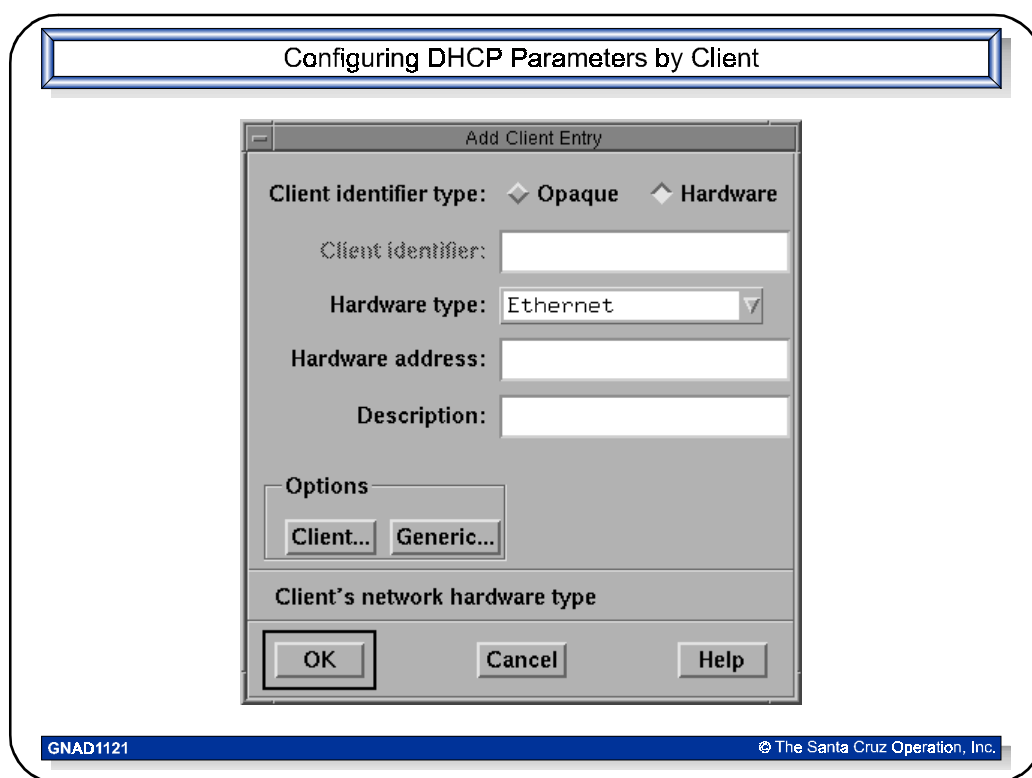


### **Exercise: Configuring DHCP subnet options**

---

Use the **DHCP Server Manager** to dynamically assign IP addresses from a pool to clients in a given subnet. Use the address pool you created in the exercise in Section 11.4.1.1, “Creating address pools,” on page 11-55.

AV 11-21



### 11.4.2.6 DHCP client options

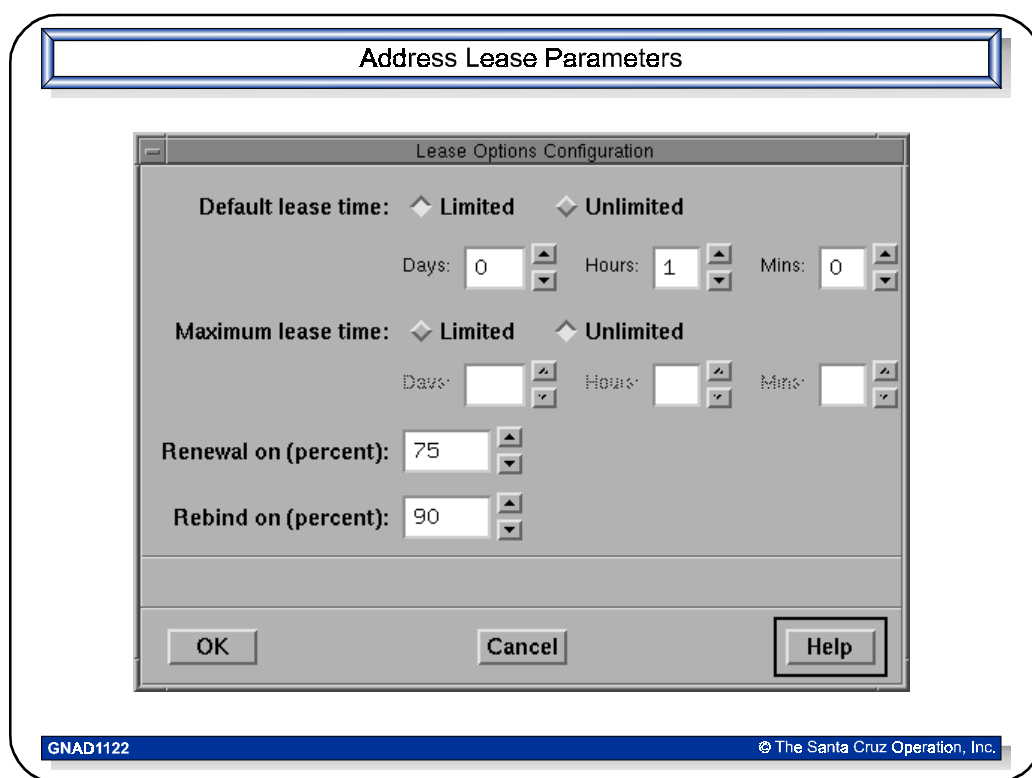
---

A DHCP client entry is often used to assign a static IP address to a particular client. This can be useful if that “client” is also acting as a server, for example sharing files or printers. In this case, it might be convenient for that system to always be assigned the same IP address so other systems on the network will know how to locate it.

To set DHCP parameters for a client with the **DHCP Server Manager**:

- Step 1.       Select **View** ⇒ **Clients**.
- Step 2.       Select **Entry** ⇒ **Add**. This displays the window shown in AV 11-21.
- Step 3.       Select a “Client identifier type” of **Hardware**.
- Step 4.       Select a “Hardware type” of either **Ethernet** or **IEEE 802**.
- Step 5.       Enter the “Hardware address” of the client. For an Ethernet address, use twelve hexadecimal digits without colons or other punctuation.
- Step 6.       Enter the “Description” for this entry. This field is optional.
- Step 7.       Set the IP address the client will receive by clicking on **Client** in the **Options** area and filling out fields in the pop-up window.
- Step 8.       Configure other parameters for this client by clicking on **Generic** in the **Options** area of the form. Use this to override global or subnet options for this client. See Section 11.4.2.4, “DHCP global options,” on page 11-63 for more information on these parameters.
- Step 9.       Click on **OK** when you are finished.

AV 11-22





### 11.4.2.7 Address lease parameters

---

When an IP address is leased, the lease can be for a limited or unlimited period of time. In addition to specifying the duration of the lease, you can specify when renewal and rebinding should occur.

To configure lease parameters from the **DHCP Server Manager**:

- Step 1. Select **View** ⇒ **Subnets**.
- Step 2. Select the subnet you want to modify.
- Step 3. Select **Edit** ⇒ **Modify**.
- Step 4. Click on **Lease** in the **Options** area of the Modify Subnet Entry window. This displays the window on the opposite page.
- Step 5. Choose a “Default lease time” of either:
  - **Limited**—the lease will expire and the client has to renew to lease or rebind to a new address.
  - **Unlimited**—the lease will not expire. If you select **Unlimited**, skip to Step 7.
- Step 6. Select the number of “Days,” “Hours,” and “Minutes” that a lease should last by default. The minimum is one hour.
- Step 7. Select the “Maximum lease time” in the same manner as the “Default lease time.” The maximum must be at least as long as the default lease time.
- Step 8. In the “Renewal on (percent)” field, indicate what percent of the duration of the lease should elapse before the client asks for a renewal. Clients should try to renew leases before they expire to allow time for the client to rebind in case the renewal is refused.
- Step 9. In the “Rebind on (percent)” field, indicate what percent of the duration of the lease should elapse before the client must rebind.

Renewal and rebinding were introduced in Section 11.4.2.1, “DHCP leases,” on page 11-59.

### 11.4.2.8 Configuring Windows as a DHCP client

---

This basic procedure outlines the configuration of a Windows workstation as a DHCP client. The specific steps used depend on the version of Windows you are using.

- Step 1.        Access network configuration screens by way of the **Network** icon in the **Control Panel**. Any installed networking components will be listed. Add the Adapter component to install a driver for your network card.
- Step 2.        Configure the Microsoft TCP/IP Protocol properties:
- Select **Obtain an IP address automatically**.
- Remove any installed gateways.
- Step 3.        Save the configuration and reboot the system.

A useful command to verify IP parameters on a Windows 95 system is **winipcfg**. It displays the currently used IP address and other network parameters. It can also be used to manually release or renew a leased IP address.

#### **Exercise: Configuring a Windows 95 client for DHCP**

---

Configure a Windows workstation to use DHCP. The specific steps for your version of Windows will be provided by your instructor.

Reboot the Windows system.

Run **winipcfg** to examine your assigned network configuration.

### **11.4.2.9 DHCP and DNS**

---

Currently, there is no good interface between DHCP and DNS. This is a problem with DHCP and dynamic addresses.

The UnixWare 7 DNS server assigns a static name to each IP address. If the IP address of a client changes, then its name must also change.

This can cause problems if:

- The client also shares resources under a given name, because that name can change with every leased IP address.
- You are using services that use the machine name for security authentication. For example, TCP wrappers, NFS, and trusted access would all be affected.

One workaround is to not use DHCP to assign IP addresses. Of course, this loses much of the functionality of DHCP. You can still use DHCP to assign such parameters as default gateway, DNS server, or domain name.

## 11.5 Lesson Summary

---

---

Today, it is rare to find a network of computers that all use the same operating system. Thus, networking means understanding cross-platform connectivity options. One model is to use a UNIX system to provide services to a network of clients running Microsoft Windows. Four of the most commonly required services are file sharing, printer sharing, access to remote applications, and remote boot configuration. For each service, there are many solutions, depending on cost, client capabilities, performance required, and other factors.

### Review questions

---

1. Which software package(s) provide file sharing and do not require the purchasing of additional software beyond UnixWare 7 and Windows 95?
2. What are two alternatives for running TermLite? What are the advantages and disadvantages of each?
3. Which alternatives for remote application access will allow a Windows user to run vi(1) on a UNIX system?
4. Which method for providing remote application access would you use to allow users to send and receive electronic mail? Why did you select that option?

5. What does the Address Allocation Server do and which other services use its service?
  
6. What relationship exists between DHCP and DNS to support dynamic binding of IP addresses to host names?
  
7. List at least three DHCP parameters that a Windows 95 machine can understand.

## 11.6 Relevant Commands, Directories, Shares, and Packages

---

The following commands, directories, shares, and packages were presented.

### Relevant commands

---

|                                       |                                                               |
|---------------------------------------|---------------------------------------------------------------|
| <code>/usr/vision/bin/visionfs</code> | VisionFS server administration tool                           |
| <code>winipcfg</code>                 | A Windows 95 command for displaying IP configuration settings |

### Relevant UNIX directories

---

|                                            |                                              |
|--------------------------------------------|----------------------------------------------|
| <code>/usr/vision/bin/pctools/en_US</code> | Home of the <b>visiontools</b> shared folder |
|--------------------------------------------|----------------------------------------------|

### Relevant VisionFS shared files

---

|                                                |                                                                            |
|------------------------------------------------|----------------------------------------------------------------------------|
| <code>visiontools\visionfs\profedit.exe</code> | The VisionFS Profile Editor                                                |
| <code>visiontools\docs\vfsintro.pdf</code>     | <b>Introducing VisionFS</b> , documentation in Acrobat format (120+ pages) |
| <code>visiontools\termlite\TermLite.exe</code> | The TermLite executable                                                    |
| <code>visiontools\termlite\Setup.exe</code>    | The TermLite installation program                                          |

### Relevant package names

---

|                       |                                |
|-----------------------|--------------------------------|
| <code>afps</code>     | Advanced File and Print Server |
| <code>nws</code>      | NetWare Services               |
| <code>termlite</code> | TermLite                       |
| <code>visionfs</code> | VisionFS                       |
| <code>winsrvcs</code> | VisionFS and TermLite          |

## 11.7 Answers to Review Questions

---

1. The software packages that provide file sharing and do not require the purchase of additional software include VisionFS and NFS. Use Vision FS if the clients are Windows 95, NT, or any other operating system that includes CIFS client support. Use NFS if the clients are other UNIX systems or any other operating system that includes NFS client support.
2. One alternative for running TermLite is to run it directly from a VisionFS shared folder. This is the easiest way, as there is no client-side installation involved. However, it creates more network traffic, as the TermLite executable must be transferred across the network every time it is used.

A second alternative for running TermLite is to install it on the Windows client. This creates less network traffic, but requires some initial installation work. It also allows you to use TermLite to connect to other systems while the VisionFS server is down.

3. Some alternatives for allowing a Windows user to run **vi** on a UNIX system are Terminal emulation and X Window servers. (The X Window server can be used to run **xterm(X1)** or **dtterm(1)**, which could in turn be used to run **vi(1)**.)
4. Some methods for providing remote application access to electronic mail include:

Using terminal emulation to access **mailx(1)**, **pine(1)**, or another UNIX text-based mail user agent.

Using an X window server to access **dtmail(1)**, **netscape(X1)**, or another UNIX X Window based mail user agent.

Using a Windows POP or IMAP client program. This is an example of a custom client agent.

5. The Address Allocation Server provides dynamic IP addresses from address pools for DHCP and PPP.
6. There is no interaction between DHCP and DNS to support dynamic binding of IP addresses to host names.

7. Windows 95 machines can understand the following DHCP parameters:

Client IP address

Netmask

IP addresses of default gateways or routers

IP addresses of DNS name servers

DNS domain name

IP broadcast address

IP address of NetBIOS name servers

NetBIOS node type

NetBIOS scope identifier

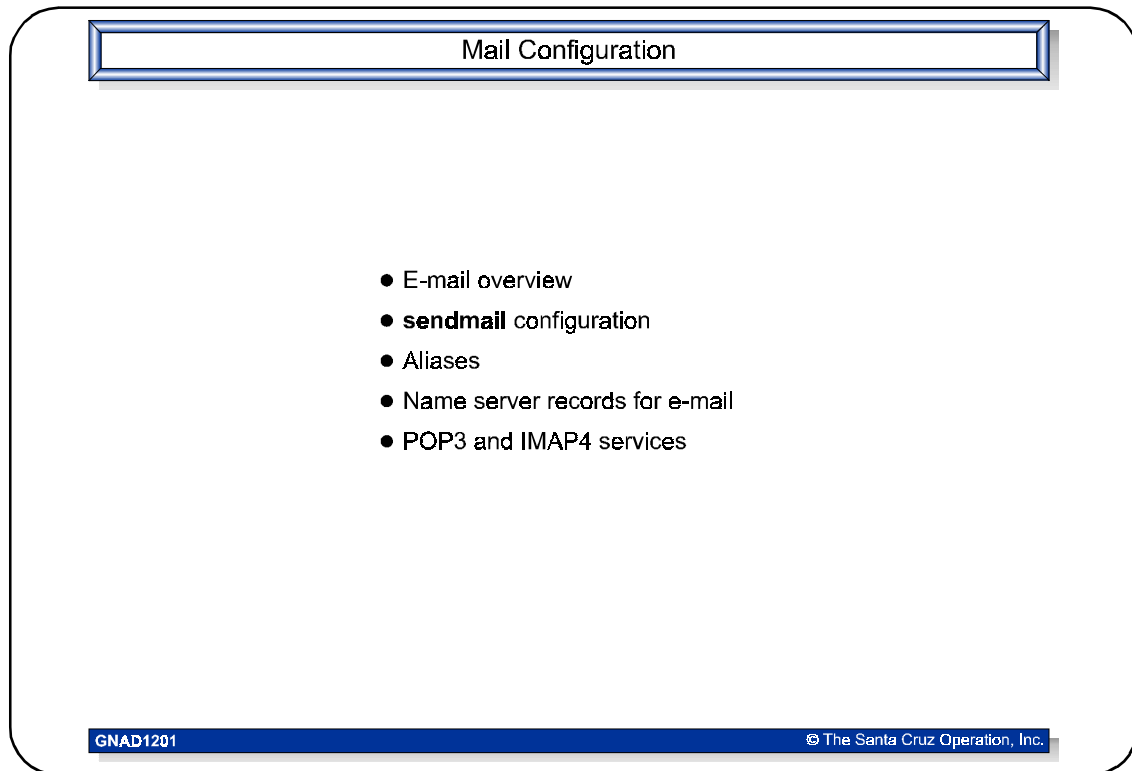


---

# ***12***

## ***Mail Configuration***

AV 12-1



**Lesson 12**

## **Mail Configuration**

### **Introduction**

---

Most of the lessons in this course focus on providing your users with seamless, effortless, and ideally, transparent access, to the resources of your enterprise network and the Internet. This chapter focuses on the administration of one of the more important tools for your users.

SCO provides a complete e-mail environment, from user interface to transfer protocols, with many configurable options. This lesson describes the various components of the SCO e-mail environment and how you can configure them for the users of your network.

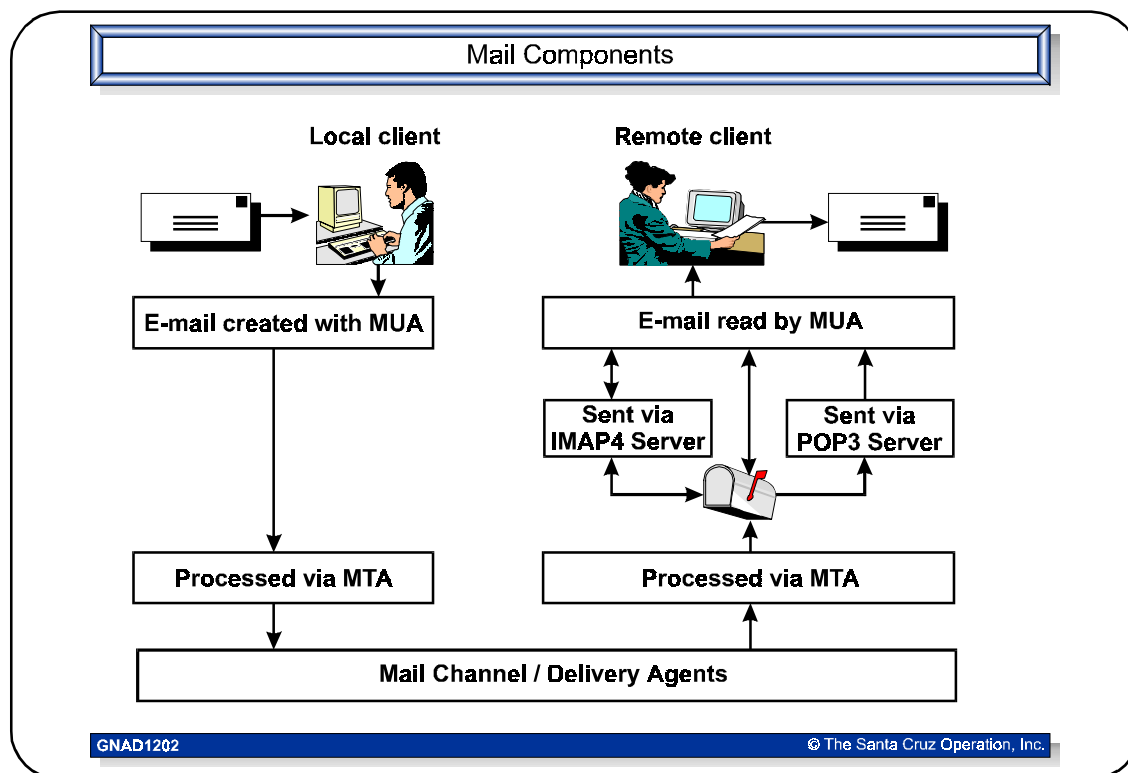
### **Objectives**

---

This lesson describes the SCO e-mail environment. By the end of this lesson, you will be able to:

- Describe the components of the e-mail system
- Configure **sendmail**(1M)
- Create global aliases and mailing lists
- Modify the name server for use with a mail transport agent
- Describe how POP3 and IMAP4 services work

## AV 12-2



## 12.1 Overview of E-mail

---

Electronic mail systems have evolved over the past several years to become a mission critical tool for many people. Systems connected to the Internet can send mail to thousands of people around the world in minutes at little or no cost.

Luckily, as part of the evolution, e-mail systems have become easier to configure with menu-driven programs. Before you start configuring e-mail, you need a thorough understanding of the various components in today's e-mail systems.

AV 12-2 shows the layers of a modern e-mail system. As with today's network protocol stacks, the software components conform to standards and are interchangeable at any layer. The layers are:

- Mail User Agent (MUA)
- Mail Transport Agent (MTA)
- Mail channels and delivery agents

AV 12-2 also illustrates where POP-3 and IMAP4 mail servers fit into the modern e-mail architecture. Each of these components are explained in this lesson.

After installation, UnixWare 7 is configured to handle local and basic Internet e-mail. If your system configuration has no special requirements, such as mail aliases, virtual domains, or UUCP connections, you might not require any further configuration of the e-mail system.

### 12.1.1 The Mail User Agent (MUA)

---

The Mail User Agent (MUA) is the program that people use to create and read their messages. MUAs are analogous to the application layer in the TCP/IP protocol stack.

Some MUAs support the Multipurpose Internet Mail Extension (MIME) format for attaching files to messages. With MIME, you can attach spreadsheets, word processing files, sound files, and any other kind of file.

Most MUAs can be configured so that when a specific MIME attachment type is selected, the MUA will automatically start a program that can understand the attachment type so that the attachment can be read, listened to, and so on. These programs are often referred to as *helper programs*.

MUAs that come with UnixWare 7 include:

**mailx(1)**      The traditional command-line mail. The UnixWare 7 implementation of **mail** supports MIME and IMAP4.

**pine(1)**      A character interface for Internet news and e-mail.

**dtmail(1X)**    An X Window System™ mail program that is included with the UnixWare 7 graphical desktop (CDE). It is also MIME capable.

**netscape(X1)** The web browser included with UnixWare 7 also includes a graphical e-mail program with MIME, POP, and HTML capabilities.

Netscape version 4 or later also supports IMAP4. This is included in release 5.0.5 but is not included in the earlier releases of UnixWare 7.

**Communicator(X1)**

An advanced Netscape web browser which includes a graphical e-mail program with MIME, POP, IMAP4, and HTML capabilities.

Other mail readers are available on the SCO Skunkware™ CD-ROM or from the Internet. One popular program is:

**elm**            A character-based menu-driven mail reader. Some versions support MIME.

There are many MUAs that can be used on Windows® desktops. Examples include Eudora®, Netscape Navigator™, and Pegasus. These MUAs will work with UnixWare 7 through a POP3 or IMAP4 server.

There are other MUAs available for Windows that are not POP3- or IMAP4-based. Such MUAs often have mail servers that act as MTAs and can gateway mail to other MTAs. Examples of these mail systems include cc:Mail and Lotus Notes.

### **12.1.2 The Mail Transport Agent (MTA)**

---

When a user sends mail, it is sent to a Mail Transport Agent (MTA) for further processing. The MTA is analogous to the IP layer in the TCP/IP protocol stack. Functions of the MTA include:

- The MTA uses the destination address(es) to determine how the message should be delivered.
- The MTA may use aliases or distribution lists to send a copy of one message to multiple recipients.
- The MTA receives and processes incoming mail from other machines on the network.

The MTA used in UnixWare 7 is **sendmail(1M)**. It is a very popular MTA that is available for most versions of UNIX.

Certain editions of UnixWare 7 include the Netscape Messaging Server™. This MTA is also available for all the editions under separate licensing agreements. The Netscape Messaging Server contains many advanced features including:

- An easy to use interface
- LDAP support, enabling access to a centralized directory of users
- Support for proprietary e-mail protocols

- Messaging quotas limiting users to fixed mailbox sizes
- Single-copy message store (messages sent within the system to multiple recipients are stored in a single shared copy)
- Filtering capabilities which can be used to reduce spam (junk-mail) from entering your server

### 12.1.3 The mail channels and delivery agents

---

The UnixWare implementation of **sendmail** is configured to use mail channels. A *mail channel* has two primary components:

- A table which determines which channel is used to deliver a mail message.
- A *delivery agent* that performs the message delivery for the channel.

The MTA identifies the first delivery channel that matches the message address in the table, and uses the associated delivery agent to transport the message. The order of the channels is significant. Any address that does not match a channel is considered undeliverable and *bounces* back to the sender.

A variety of channels are prebuilt on UnixWare 7.

Those activated by default are:

- The local channel, which handles delivery of mail to local users.
- The SMTP channel, which delivers mail over TCP/IP networks using DNS and the Simple Mail Transport Protocol (SMTP).
- The badhost delay channel, which by default queues messages that temporarily fail because the name server is not available. This channel can also be configured to forward e-mail for unknown hosts to a *smarter* mail host for delivery.



Those available for activation are:

- The UUCP channel, which delivers mail to hosts configured as UUCP connections in the */etc/uucp/Systems* file.
- The baduser channel, which delivers mail destined for unknown users to a smarter machine. This smarter machine may be used to centralize a company alias table and reduce administration.
- The multihome channel, which is used for routing mail for virtual domains. Virtual domains allows a single host to manage mail for multiple domain names.

The associated delivery agents used by these channels are:

|                             |                                                                                                                                 |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>/etc/mail/slocal</i>     | A program that performs local mail delivery. See <b>slocal</b> (1).                                                             |
| SMTP                        | <b>sendmail</b> has built in support for SMTP so no external program is necessary.                                              |
| <i>/usr/lib/uucp/uucico</i> | The Unix-to-Unix copy system is used to transfer mail for UUCP channels.                                                        |
| <i>/etc/mail/multihome</i>  | A program that routes mail for virtual domains to the correct user account.                                                     |
| Custom channel              | A custom channel available for configuration. Any pathname can be specified but the default path begins with <i>/etc/mail</i> . |

## 12.1.4 POP and IMAP servers

---

The Post Office Protocol (POP) and Internet Message Access Protocol (IMAP) daemons serve as an intermediary between the MUA and the MTA, delivering mail on a per-user basis. By contrast, an MTA, such as SMTP, delivers mail on a per-system basis.

POP and IMAP work as follows:

- Incoming mail is received by the MTA and stored in a file on the UNIX server.
- The user runs a MUA that is a POP or IMAP client. This MUA may run under Windows, UNIX, or almost any other operating system. Note that the MUA need not run on the same computer as the MTA.
- The POP or IMAP client downloads newly arrived e-mail from the mail server.
- The user can now disconnect their computer from the network, if desired, and read their mail off-line.
- Outgoing mail is sent via SMTP to the UNIX server or the final destination, depending on client configuration.

For example:

- You might take a notebook computer on a trip.
- While waiting for an airplane, you turn on the notebook, plug the modem into a phone, dial up, and download e-mail via either POP or IMAP.
- While on the plane, you read e-mail, formulate replies, and create new messages.
- These messages are saved on the notebook until the next time you connect to the mail server and download e-mail again (that is, the responses are uploaded at the same time).

The primary differences between POP and IMAP are:

- POP allows only one way transfer from a central mail server to the client.
  - POP is usually configured to delete your messages from the server when they are downloaded.

It can be configured so the downloaded messages are not deleted off the server. This maybe beneficial when using POP from multiple sites for the same e-mail account.

- With POP, old e-mail is stored on the client machine, not the server.
- If you read your mail from a new client, you do not have access to your saved messages, and new mail you receive is saved on the new client.
- IMAP allows two way transmissions.
  - Your saved messages are stored in folders on the server.
  - You can manipulate your mail off-line, and when you next have a network connection to the server, your changes are uploaded and applied.

This allows you to:

- Delete mail messages
- Save messages to a folder
- Move messages between folders

UnixWare 7 comes with a POP version 3 (POP3) mail server and an IMAP version 4 (IMAP4) mail server. These daemons are started by default by the **inetd(1M)** daemon. As previously discussed, to take advantage of a POP3 or IMAP4 server, you need a POP3 or IMAP4 client.

## 12.2 Configuring sendmail

---

Configuring a **sendmail**-based mail system can involve one or more of these functional components:

|                 |                                                                                                                                                          |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Mail router     | Basic mail router functionality is controlled by the <i>/etc/sendmail.cf</i> file. This file stores information about the server and mailers             |
| Folders         | Directories used to store mail messages maintained as folders.                                                                                           |
| Aliases         | Aliases are used by <b>sendmail</b> to process mail.                                                                                                     |
| Name resolution | Special DNS records are used to route mail destined for remote systems on a TCP/IP network.                                                              |
| User options    | Users can customize mail options that affect only their own incoming or outgoing mail. Examples are vacation notification and forwarding to other users. |

These components, and the tools used to configure each feature, are discussed in more detail in the next sections.

### 12.2.1 Default sendmail configuration

---

On UnixWare 7, **sendmail** is preconfigured to support mail delivery on the local system and the LAN. For many situations, no additional **sendmail** configuration is required. Mail is delivered as follows:

- Incoming mail:
  - Delivered locally if the destination user exists
  - Rejected and returned to the sender if the destination user does not exist on this system
  - Forwarded to the remote system if destined for another system (indicated by *@hostname.domain* in the “To:” line)
  - Mail to this system’s *postmaster* is delivered to **root**

- Outgoing mail:
  - Delivered directly to the system in the address
  - Append the fully-qualified domain name to the sender's user name

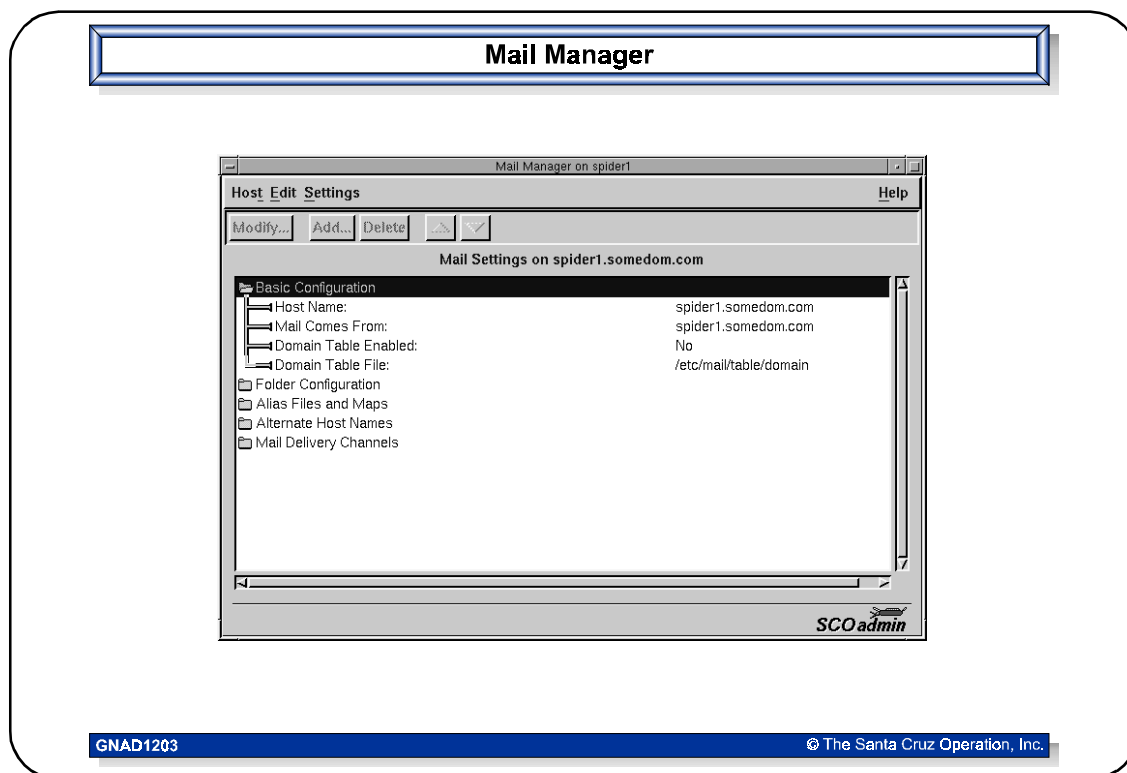
### **Exercise: Default sendmail configuration**

---

Use the default mail configuration to send and receive e-mail:

- |         |                                                                                                                                                                  |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Step 1. | Create two user accounts.                                                                                                                                        |
| Step 2. | Log in on multiscreens as each user.                                                                                                                             |
| Step 3. | Use <b>mail</b> (1) or <b>dtmail</b> (1X) to send a message between the two users. Verify that it is delivered.<br><br>Note the domain name in the "From:" line. |
| Step 4. | Send mail to a non-existent user on your local system. Check your mailbox. What happens to this message?                                                         |
| Step 5. | Send mail to an existing account another system on the LAN. Verify that was delivered.                                                                           |

AV 12-3



## 12.2.2 Using the Mail Manager to modify sendmail

---

To configure these basic **sendmail** options:

Step 1. Run **scoadmin**(1M).

Step 2. Select **Mail Manager**.

Step 3. Double click on the **Basic Configuration**.

Step 4. Double-click on each option as needed:

|                      |                                                                                                                                                                                                                                                                                                               |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Host Name            | Indicates the host name used by the mail system. In general, this will not need to be modified unless you have multiple interfaces and <b>sendmail</b> chooses the wrong one.                                                                                                                                 |
| Mail Comes From      | Indicates the host name used in the return address of outgoing mail messages. This can be used to configure <i>masquerading</i> or <i>host hiding</i> . For example, you may want mail sent from <b>user@foo.sco.com</b> to appear as if it was sent from <b>user@sco.com</b> .                               |
| Domain Table Enabled | Indicates whether domain tables are enabled. A domain table allows mapping of a short host name into a fully qualified domain or indicating a route for sending mail to a specified domain. See <b>scohelp</b> ⇒ <b>Mail and Messaging</b> ⇒ <b>Administering Mail and Messaging</b> ⇒ <b>Domain tables</b> . |
| Domain Table File    | Indicates the name of the domain table file.                                                                                                                                                                                                                                                                  |

### **Exercise: Configuring host hiding**

---

Use the **Mail Manager** to remove the system name in "Mail comes from:" field.

Send mail between two user accounts and note the name in the "From:" line.

Further configuration of **sendmail** must be done by manually editing the *sendmail.cf* file. See Section 12.5, "Configuring Sendmail Manually," on page 12-23.



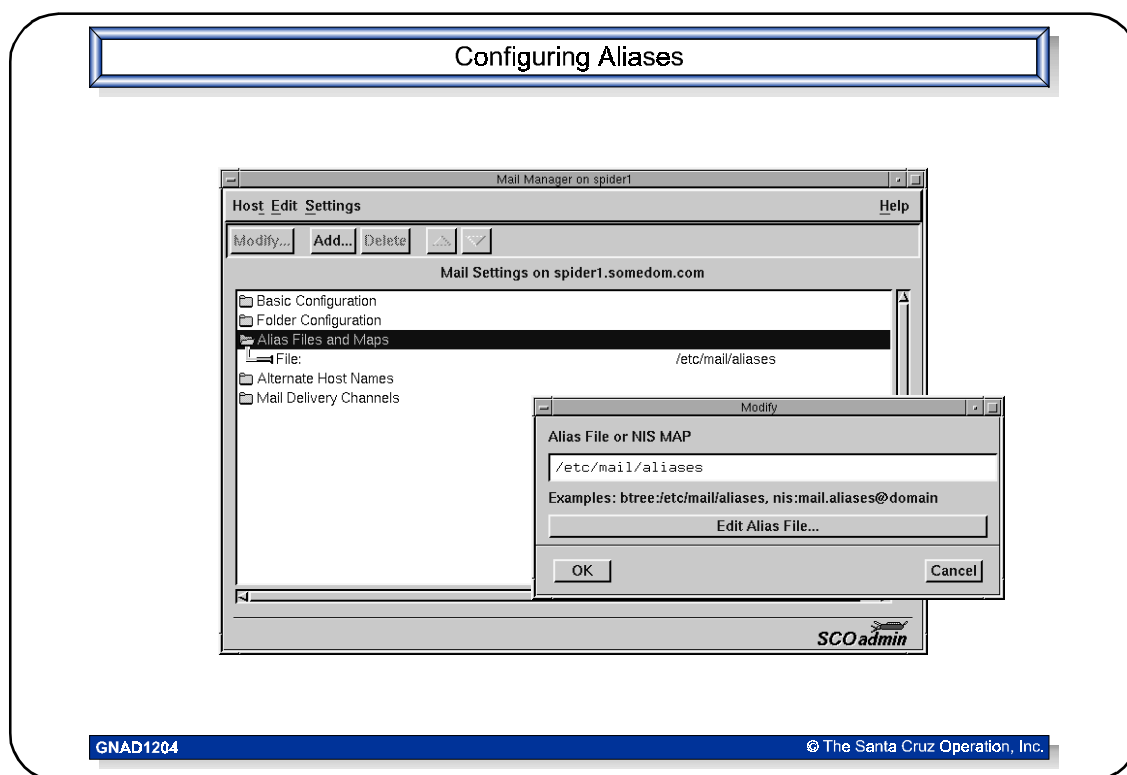
## 12.3 Configuring Folders

---

The **Folder Configuration** folder has the following entries:

|                                     |                                                                                                                                                                                                                                                                                      |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Users' INBOX Location               | Configure the location of the mail store by choosing between the System Spool Directory, the Users' Home Directory, or a custom location. The location you select should be on a filesystem that has plenty of free space.                                                           |
| Default Folder Format               | Choose either sendmail or MMDF for the default format. Unless you need backwards compatibility with SCO OpenServer™, choose <b>sendmail</b> .                                                                                                                                        |
| Folder Synchronized to Disk         | If you select synchronize folders to disk, reliability is improved in case of a system crash, but throughput is decreased.                                                                                                                                                           |
| Folder Consistency Checking         | Select this option only if you use a MUA program that modifies folders without changing the size or time stamp of the file.                                                                                                                                                          |
| Folders Cached in Memory            | Caching folders in memory may improve performance, but less memory is left for other system functions.                                                                                                                                                                               |
| Folder Garbage Collection Threshold | You can specify how often to remove deleted messages from a folder. The threshold is a percentage, where 0% prevents garbage collection from ever occurring, and 100% means garbage collection occurs immediately. Lower values mean better performance, but more wasted disk space. |
| Folder Lock Timeout                 | You can specify the length of time <b>sendmail</b> waits for access to a mail folder.                                                                                                                                                                                                |
| File Based Locking Enabled          | Enable file-based locking if you have MUAs that do not use kernel-based locking.                                                                                                                                                                                                     |
| File/Directory Creation Umask       | Specify the file creation umask for mail folders.                                                                                                                                                                                                                                    |

## AV 12-4



## 12.4 Configuring Aliases

---

Aliases are used to:

- Allow mail to be sent to one user under a variety of different user names, or one address to be redirected to another
- Create mailing lists
- Append mail to files, possibly for archival purposes
- Filter mail through programs or shell scripts

For each alias file, a pathname to the alias file is specified. Each file may contain many aliases. The default alias file is */etc/mail/aliases*.

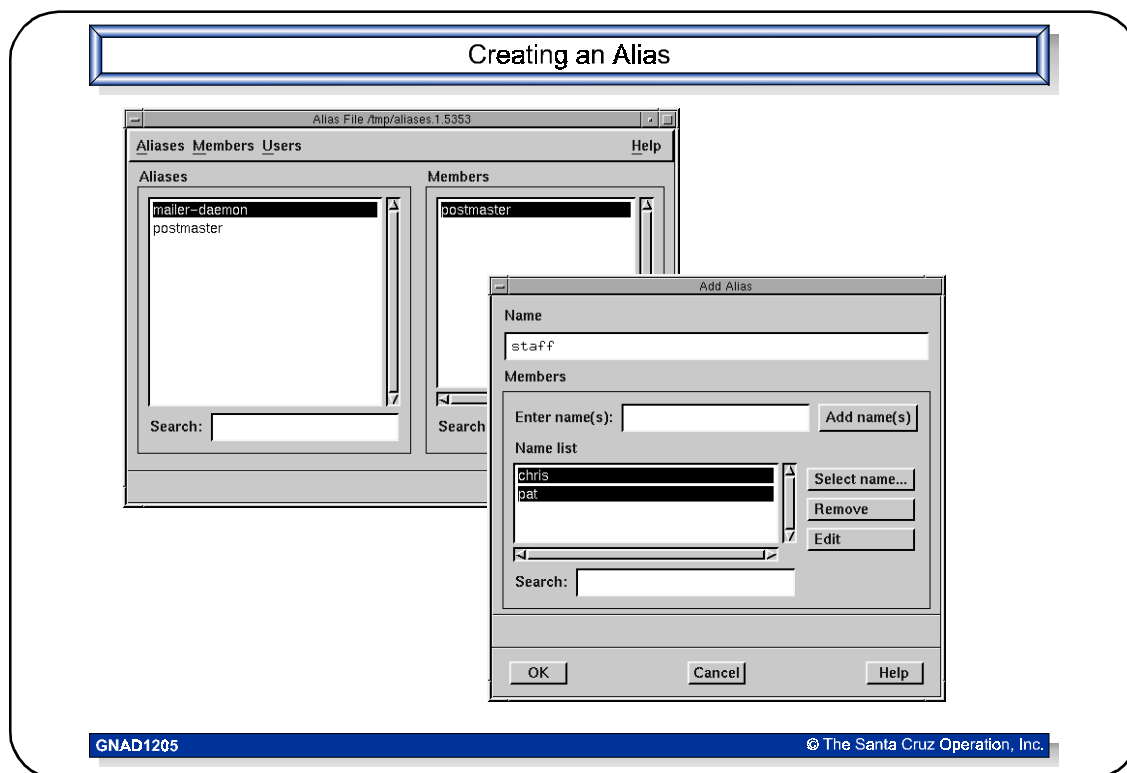
### 12.4.1 Modifying the default alias file

---

To add an alias to the default alias file:

- Step 1. Start the **Mail Manager**.
- Step 2. Double-click on **Alias Files and Maps**.
- Step 3. Select the default file, */etc/mail/aliases*.
- Step 4. Select **Edit** ⇨ **Modify**.
- Step 5. Click on **Edit Alias File**.

## AV 12-5



- Step 6.        Select **Aliaes** ⇨ **Add**. Complete the form:
- |               |                                                                                        |
|---------------|----------------------------------------------------------------------------------------|
| Name          | Enter a name for the alias. This is the name used to send mail to this group of users. |
| Enter name(s) | Enter a list of user names, separated by spaces.                                       |
- Step 7.        Click on **Add name(s)**.
- Step 8.        Select **Aliases** ⇨ **Exit** when finished.
- Step 9.        Click on **Yes** to save the changes.
- Step 10.       Click on **OK** to return to the **Mail Manager**.

**Exercise: Configuring Aliases**

---

---

Modify the default alias file to create a new alias named **staff** for the two users created earlier in this lesson.

Send mail from **root** to **staff**. Verify that both users receive the mail.

## 12.4.2 Alias subscription

---

When you create new user accounts, you may want to *subscribe* the user to a number of aliases at once:

- Step 1. Start the **Mail Manager**.
- Step 2. Double-click on **Alias Files and Maps**.
- Step 3. Select the default file, */etc/mail/aliases*.
- Step 4. Select **Edit** ⇨ **Modify**.
- Step 5. Click on **Edit Alias File**.
- Step 6. Select **Users** ⇨ **Subscribe**.
- Step 7. Select a name from the list (*/etc/passwd*) or enter a user name. Click on **OK**.
- Step 8. Select the aliases to which you want this user to belong. Click on **Add**.
- Step 9. Click on **OK** when finished.
- Step 10. Select **Aliases** ⇨ **Exit** when finished.
- Step 11. Click on **Yes** to save the changes.
- Step 12. Click on **OK** to return to the **Mail Manager**.

## 12.4.3 Mail Delivery Channels

---

Mail Delivery Channels were discussed in Section 12.1.3, “The mail channels and delivery agents,” on page 12-8. The **Local**, **SMTP**, and **Badhost** channels are configured by default. The remaining channels can be configured by following these steps:

- Step 1. Select **Mail Delivery Channels**.
- Step 2. Select **Edit** ⇨ **Add** from the menu.

## 12.5 Configuring Sendmail Manually

---

The configuration tools presented in the previous section will likely be sufficient for making changes to your **sendmail** configuration. However, these tools do not always have the functionality you need. If this is the case, you can attempt to directly modify the configuration files.

If you plan to manually edit the configuration files, first use the **Mail Manager** to do as many of the configuration changes as possible before manually configuring **sendmail**.

### 12.5.1 The *sendmail.cf* file

---

The main configuration file for the mail system is */etc/sendmail.cf*. For information regarding the structure and modification of this file, read:

Costales, Bryan with Eric Allman, *sendmail Second Edition*, Sebastapol, CA: O'Reilly & Associates, 1997, ISBN 1-56592-222-0

The message store for the mail system is configured in the */etc/default/mail* file. See the manual page *mail(4)* for information on this file.

One possible use for direct manipulation of the configuration files is to configure multiple replicated systems. Instead of repeating the same configuration setup on each system, the configuration file can be copied from one system to another.

## 12.5.2 Alias files

---

The mail system can be configured to use a variety of alias files, each using a different database format. By default, the ASCII text file */etc/mail/aliases* is the only alias file. This file can be modified directly using a text editor. For more information, see the *aliases*(4) manual page. Entries in the file take the form:

`aliasname:address[, address]`

where *address* is any of:

- A local user name
- A standard Internet e-mail address, e.g. *user@foo.com*
- The pathname of a file, to which messages are appended
- A file listing e-mail addresses, one per line, to be included in this mail alias, e.g. `:include:/etc/mail/staff`
- A command, to which messages are piped, e.g. `"| lp -d laser1"`
- A UUCP e-mail address, e.g. *host1!host2!user*

An example *aliases* file can include these entries:

```
matt: matthew
peter: peter@admin.somedom.com
info: kathy, janice, vicki
diary: /var/adm/log/diaryfile
staff: :include:/etc/mail/staff
printer: "| lp -d laser1"
ken: host1!host2!ken
```

After editing the */etc/mail/aliases* file, you must run the command **newaliases**(1M) to rebuild the **sendmail**(1M) alias database. The syntax is:

**`/etc/mail/newaliases`**



## 12.6 User-configurable mail options

---

There are several features of **sendmail** that users can take advantage of without the intervention of the system administrator. These include:

- Forwarding mail to another user account using the *.forward* file
- Automatic responses to incoming mail during absences
- Processing incoming mail using the *.maildelivery* file

### 12.6.1 *.forward* files

---

When users move from one location to another, often they continue to receive mail at their old address. To have the mail forwarded to a new address or user name, put a *.forward* file in the user's home directory, indicating where their mail should be sent.

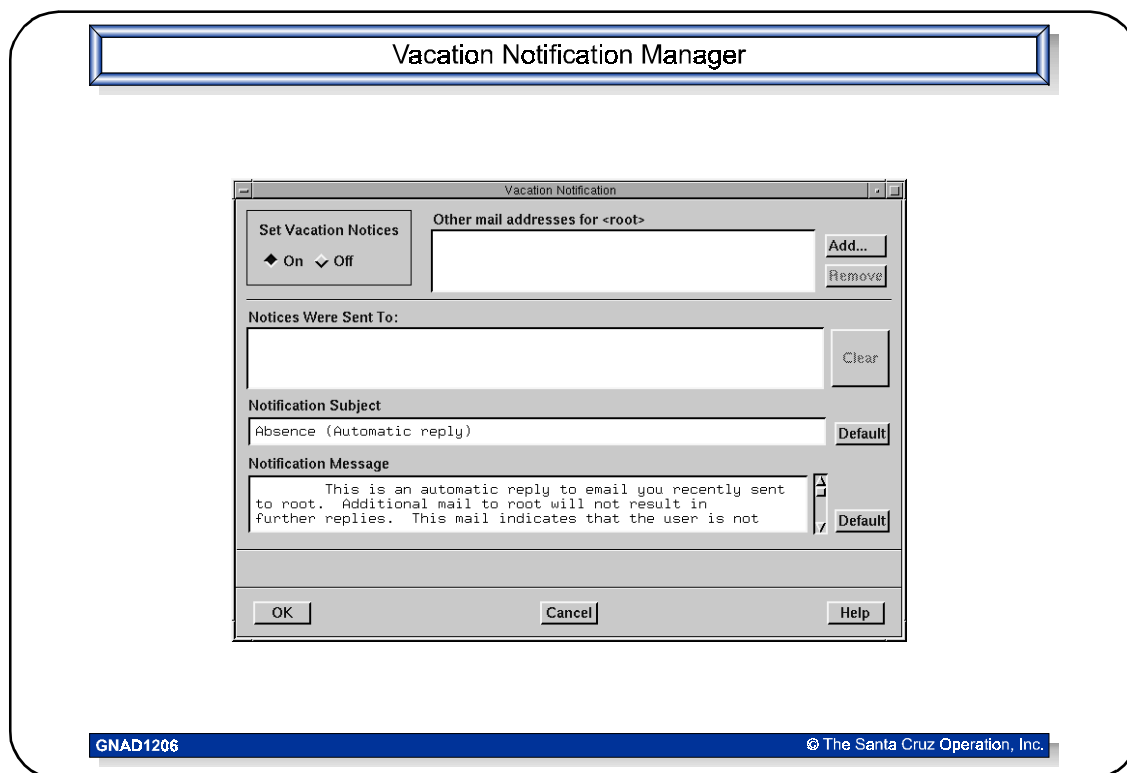
The format of the *.forward* file is a single e-mail address on one line. For example:

```
donna@machine2.somedom.com
```

When entered in the *.forward* file in user **donna**'s home directory on *machine1*, all mail for **donna** on *machine1* will go to **donna** on *machine2.somedom.com*.

You should be careful not to configure two machines to redirect mail to each other, as this creates an endless mail loop.

## AV 12-6



## 12.6.2 Extended absence notification

---

Even when you are unable to read e-mail during extended absences, people will likely continue to send you e-mail. You can have the mail subsystem send a notification to people who send you messages to let them know that you are out of the office.

UnixWare 7 has the **Vacation Notification Manager** for this purpose. This manager does not require **root** privileges so ordinary users use it to configure absence notices.

To configure vacation notification:

- Step 1.       Enter the command from a graphical terminal window:  
  
              **/etc/mail/admin/vacation**
- Step 2.       Click the **On** button in the “Set Vacation Notices” section.
- Step 3.       Enter your reply message in the “Notification Message” field.
- Step 4.       Click on **OK**.

.

### **Exercise: Configuring Vacation Notification**

---

Configure your system as if you were on vacation. Write an appropriate notification message.

### 12.6.3 Maildelivery Files

---

Your inbound mail can be processed according to a complex set of selection criteria, specified in the file *.maildelivery* in the your home directory. Each line of this file consists of five fields, separated with spaces. The format of each line in the *.maildelivery* file is:

header pattern action result string

where:

|         |                                                                                                                                                                                                                                                                                                       |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| header  | The message header line that is searched                                                                                                                                                                                                                                                              |
| pattern | The text to look for in the indicated header line of the message                                                                                                                                                                                                                                      |
| action  | One of: <ul style="list-style-type: none"><li>• <b>file</b>—append the message to a file</li><li>• <b>pipe</b>—pipe the message to an executable program</li></ul>                                                                                                                                    |
| result  | One of: <ul style="list-style-type: none"><li>• <b>A</b>—if the above action succeeded, then the message is considered delivered and will not be added to the user's mailbox</li><li>• <b>R</b>—deliver the message to the user's mailbox regardless of whether the above action succeeded.</li></ul> |
| string  | The file or command that is used                                                                                                                                                                                                                                                                      |

For example, the following line will cause all mail addressed to *mail\_list@otherdom.com* to be appended to the file *list.mail* in the user's home directory:

```
To mail_list@otherdom.com file A $HOME/list.mail
```

For another example, this line will take all mail with the subject *printme* and send it to the printer *laser1*:

```
Subject printme pipe A /usr/bin/lp -d laser1
```

For more information see the *maildelivery(4)* manual page.

## 12.7 Mail and DNS

---

If your network is attached to the Internet, you might need to add records to your DNS server to support your e-mail system. DNS uses a special record type, MX records, for identifying mail exchangers on the Internet.

When a message is to be sent over a TCP/IP network, the MTA queries the DNS system to determine if there is an MX record for the destination name.

If no MX record is found for that name, most MTAs will try delivering the message to a host with the same name. However, some MTAs do not do this. Therefore, if you do not specify an MX record for your host, some servers might be unable to reach you.

Typically, in an MX record, you list your mail gateway and one or two machines outside of your organization that are willing to queue mail for you if your Internet connection goes down.

On the master name server, add MX records to define the default mail exchangers for a host:

- Step 1. Start the **DNS Manager**.
- Step 1. Select a zone. Select **View** ⇨ **Records**.
- Step 2. Select a host.
- Step 3. Select **Records** ⇨ **Modify**.
- Step 4. Click on **Mail exchangers**.
- Step 5. Click on the "Mail exchanger" field and enter the name of the mail exchanger that this host can use.
- Step 6. Click on **Add**.
- Step 7. Click on **OK** until you return to the main **DNS Manager** window.
- Step 8. Click on **Host** ⇨ **Exit** to exit the **DNS Manager**.

Another use for MX records is to reroute mail for a host name to another host. The first host name does not even have to be the name of a valid host on your network and often is simply the domain name.

To see why this is useful, consider the following situation:

- All e-mail addresses in your company are to appear in the form *user@somedom.com*.
- *somedom.com* is not a valid host name for a machine as it is the domain name.
- Users in the company may be using a variety of different machines.

To implement this scenario:

- Create MX records for the name *somedom.com*.
- Set up host hiding on the mail server. This causes the mail sent out to have the proper reply address; for example *ralph@somedom.com*.
- Configure forwarding on all the client machines to use this mail server and to send all outgoing messages with the proper reply address.
- Create aliases to tell the MTA where to send incoming mail.

See **scohelp** ⇒ **Internet and Intranet** ⇒ **Configuring Domain Name System (DNS) servers** for more information on DNS configuration.

## 12.8 Client-Server Mail (POP and IMAP)

---

POP and IMAP mail were introduced in Section 12.1.4, “POP and IMAP servers,” on page 12-10. The POP3 server */etc/popper* and the IMAP4 server */etc/imapd* are already configured to be run by **inetd**(1M) when **inetd** detects POP3 or IMAP4 requests.

Using these services requires no configuration on the server.

To disable these servers, edit */etc/inet/inetd.conf* and comment out the lines containing */etc/imapd* and */etc/popper*:

```
#pop3 stream tcp nowait root /etc/popper
#imap4 stream tcp nowait root /etc/imapd
```

Next, find the **inetd** process ID and signal the **inetd** process to re-read this file:

```
ps -ef | grep inetd | grep -v grep
kill -1 pid
```

Most POP3 and IMAP4 clients need to know a minimum of information about the server:

- The name of the host running the POP3 or IMAP4 server
- The username of the account to which the mail is delivered
- The password of the account for which the client is to retrieve mail

When using POP and IMAP, it is possible to configure e-mail accounts on your system for users that are not allowed terminal login privileges. To do this, configure their accounts to use */bin/false* as their login shell.

Even though they cannot login to the system, they can still access their e-mail through remote clients using either POP or IMAP.

## 12.9 Lesson Summary

---

---

This lesson described the components of a full e-mail environment and presented the information necessary for you to configure **sendmail**, including modifications to DNS configuration.

### Review questions

---

1. What are the three layers in an e-mail system?
2. What do you have to do initially to enable e-mail on your UnixWare 7 system?
3. Which DNS record designates hosts as mail exchangers?
4. What must be done to the *inetd.conf* file to disable POP3 services under UnixWare 7?



## 12.10 Relevant Commands and Files

---

The following commands and files were presented in this lesson.

### Relevant commands

---

|                     |                                              |
|---------------------|----------------------------------------------|
| <b>dtmail(1X)</b>   | X Window System mail program included in CDE |
| <b>imapd(1M)</b>    | Daemon that handles IMAP mail access         |
| <b>mailx(1)</b>     | Traditional UNIX command-line mail program   |
| <b>popper</b>       | Daemon that handles POP mail access          |
| <b>sendmail(1M)</b> | Standard UnixWare 7mail transfer program     |
| <b>vacation</b>     | Configures extended absence notifications    |

### Relevant files

---

|                             |                                                                          |
|-----------------------------|--------------------------------------------------------------------------|
| <i>/etc/inetd.conf</i>      | Internet services configuration file                                     |
| <i>/etc/mail/slocal</i>     | Program that performs local mail delivery                                |
| <i>/etc/mail/multihome</i>  | Program that routes mail for virtual domains to the correct user account |
| <i>\$HOME/.forward</i>      | File which redirects mail to the address listed in the file              |
| <i>\$HOME/.maildelivery</i> | File which configure automatic handling of incoming mail                 |
| <i>/etc/sendmail.cf</i>     | Configuration file for <b>sendmail(1M)</b>                               |
| <i>/etc/mail/aliases</i>    | Default location for system mail aliases                                 |
| <i>/usr/lib/uucp/uucico</i> | Unix-to-Unix copy system is used to transfer mail for UUCP channels      |

## 12.11 Answers to Review Questions

---

---

1. The three layers in a e-mail system are the mail user agent (MUA), mail transfer agent (MTA), and mail channel / delivery agents.
2. Nothing needs to be done to enable e-mail. Once installed and hooked to the network, UnixWare is preconfigured to send and receive e-mail.
3. Add MX records using the **DNS Manager** to indicate which hosts act as mail exchangers for your network.
4. To disable the POP3 server, edit */etc/inetd.conf* and comment out the line with */etc/popper*. Then, signal the inetd process with **kill -1**.



***AT74401P001***