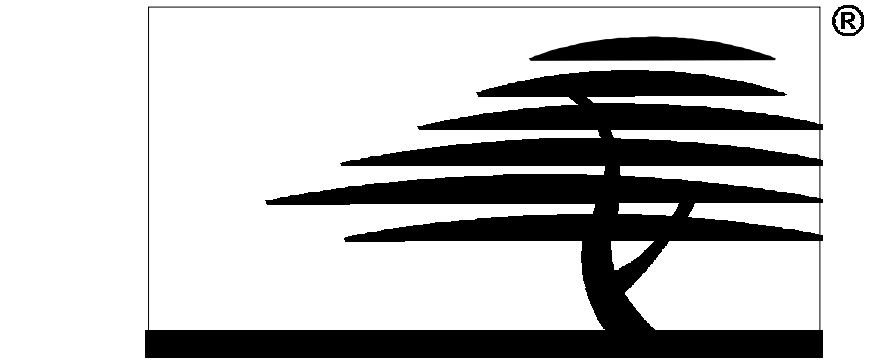


SCO

SCO OpenServer™ Release 5 Network Administration

Instructor Guide

Course Release: 2.0.1

For Software: SCO OpenServer™
Release 5

©1983-1999 The Santa Cruz Operation, Inc. All rights reserved.

©1984-1990 AT&T.

This publication is protected under copyright laws and international treaties. ©1992-1994 AT&T Global Information Solutions Company; ©1987-1989 Computer Associates, Inc.;©1987 Convergent Technologies, Inc.; ©1989 Digital Equipment Corporation; ©1987-1989 Hewlett-Packard Company; ©1994-1996 Netscape Communications Corporation; ©1993-1994 PlatinumTechnologies; ©1988 Massachusetts Institute of Technology; ©1985-1989 Metagraphics SoftwareCorporation; ©1980-1989 Microsoft Corporation; ©1989 Open Software Foundation, Inc.; ©1993-1994 Programmed Logic Corporation; ©1994-1997 Sun Microsystems, Inc.; ©1988 UNIX Systems Laboratories, Inc.; ©1997 VERITAS Software Corporation.

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.

Business/21, dbXtra, DiSCOver, Internet Way of Computing, IWoC, Multiscreen, ODT, Open Desktop,Optimized For Internet Computing and its logo, Panner, SCO, SCO ACE, SCO CIFS Bridge, SCODoctor, SCO Doctor for Networks, SCO Doctor Lite, SCO Global Access, the SCO logos, SCO MPX, SCO MultiView, SCO Nihongo OpenServer, SCO OK, the SCO OK logo, SCO OpenServer, SCO Open Server, SCO Portfolio, SCO POS System, SCO Premier Motif, SCO TermLite, SCO ToolWare,SCOtopia, SCO Vision 97, SCO VisionFS, SCO Visual Tcl, Skunkware, Tarantella, The Business Choice,The Santa Cruz Operation, UnixWare, Universal Server, VP/ix, and Zones are trademarks or registered trademarks of The Santa Cruz Operation, Inc. in the USA and other countries. APC, SoftCare, and SoftTech are service marks of The Santa Cruz Operation, Inc. Deskterm, Deskworks, IXI, IXI Desktop, the IXI logo, IXI Panorama, Wintif, and X.desktop are trademarks or registered trademarks of IXI Limited, a subsidiary of The Santa Cruz Operation, Inc. X.tra is a service mark of IXI Limited. Codon, Devkit.Vision,Esprit, Kodon, PC-Connect, SQL-Retriever, SuperVision, Super.Vision, TermVision, Term.Vision, Vision Builder, Visionware, Visionware Direction, the Visionware logo, Visionware SQL-Retriever, Visionware Super.Vision, the XV logo, XVision, and X-Visionware are trademarks or registered trademarks of Visionware Limited, a subsidiary of The Santa Cruz Operation, Inc. X/Open and UNIX are registered trademarks and the X Device is a 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. Adobe is a trademark of Adobe Systems Incorporated and is registered in the U.S. Patent and Trademark Office. Reliant is a registered trademark of Siemens Pyramid Information Systems, Inc. (formerly Pyramid Technology Corporation).

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 Santa Cruz Operation, Inc. reserves the right to change or modify any of the product or service specifications or features described herein without notice.

This document is for information only. SCO makes no express or implied representations or warranties in this document.

The Santa Cruz Operation, Inc. and SCO Skunkware are not related to, affiliated with or licensed by the famous Lockheed Martin Skunk Works®, the creator of the F-117 Stealth Fighter, SR-71, U-2, Venturestar , Darkstar, and other pioneering air and spacecraft.

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: When licensed to a U.S., State, or Local Government, all Software produced by SCO is commercial computer software as defined in FAR 12.212, and has been developed exclusively at private expense. All technical data, or SCO commercial computer software/documentation is subject to the provisions of FAR 12.211 - "Technical Data", and FAR 12.212 - "Computer Software" respectively, or clauses providing SCO equivalent protections in DFARS or other agency specific regulations. Manufacturer: The Santa Cruz Operation, Inc., 400 Encinal Street, Santa Cruz, California 95060.

The copyrighted software that accompanies this manual is licensed to the End User only for use in strict accordance with the two End User License Agreements. All such Agreements should be read carefully before commencing use of the software. Document Version: UnixWare 7

30 January 1998



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 ... 1-3
- Course goal and purpose ... 1-3
- Target audience ... 1-4

Student Materials ... 1-7
- The ***Student Guide*** ... 1-7
- SCO documentation ... 1-7

Typographical Conventions ... 1-9
Using **scoadmin**(ADM) ... 1-11
Sample exam items ... 1-13

Lesson 2: TCP/IP Concepts **2-1**

Hardware Addresses ... 2-7
Software Addresses ... 2-11
- IP addresses ... 2-11
- Netmasks ... 2-19
- Network addresses ... 2-20
- Broadcast addresses ... 2-20
- Available host addresses ... 2-23
- Multicast addresses ... 2-23

Network-Joining Devices ... 2-27
Host Names and IP Addresses ... 2-29
- System name ... 2-29
- Domain names ... 2-29
- The */etc/hosts* file ... 2-33

Translating IP Addresses to Hardware Addresses ... 2-41
- Address resolution protocol (ARP) ... 2-41
- The ARP cache and the **arp**(ADMN) command ... 2-41

Protocols and Ports ... 2-43
- Clients and servers ... 2-43
- Ports ... 2-43
- Transport protocols ... 2-44

The TCP/IP Protocol Stack ... 2-46
- TCP/IP layers ... 2-49
- Transmission between the layers ... 2-51
- An example transmission ... 2-51

LAN Hardware ... 2-54
- Ethernet ... 2-54
- IBM Token Ring ... 2-57
- FDDI ... 2-59
- ATM ... 2-59

The Future of the Internet Protocol (IPv6) 2-60
Lesson Summary 2-61
Relevant Commands, Files, and Syntax 2-63
Answers to Excercises and Review Questions 2-64

Lesson 3: Configuring a Network Adapter 3-1

Configuring Network Interfaces 3-6
Network hardware 3-6
Network adapter drivers 3-9
Testing TCP/IP Connectivity 3-15
The **netstat**(TC) command 3-15
The **ping**(ADMN) command 3-16
The `loopback` interface 3-17
Verifying network hardware 3-21
Modifying Network Interface Components 3-23
Adding protocols 3-23
Modifying and removing components 3-24
Lesson Summary 3-25
Lab: Configuring a Network Node 3-26
Configuring a network adapter 3-26
Relevant Commands and Files 3-30
Answers to Excercises and Review Questions 3-31

Lesson 4: Subnets 4-1

Planning a Subnet Topology 4-6
Increasing the number of network nodes 4-9
Joining multiple LANs 4-9
Increased network traffic 4-11
Creating Subnets 4-13
Dividing up the network 4-13
Class A subnets 4-17
Class B subnets 4-25
Class C subnets 4-31
Unusual subnet masks 4-39
Lesson Summary 4-40
Relevant Commands and Files 4-42
Answers to Excercises and Review Questions 4-43

Lesson 5: Routing 5-1

Configuring a System as a Router 5-7
Installing additional network adapters 5-9
Kernel parameters 5-11
Configuring Routes 5-15
Direct and indirect routing 5-15

The routing table 5-17
The **netstat(ADMN)** command 5-19
Updating routes 5-20
Manual updates and **route**(ADMN) 5-23
Default routes 5-25
Dynamic Routing Table Updates and Routing Daemons 5-28
routed(ADMN) 5-31
gated(ADMN) 5-36
irdd(ADMN) 5-37
Troubleshooting Routes 5-38
The **ping**(ADMN) command 5-38
The **traceroute**(ADMN) command 5-39
The **ripquery**(ADMN) command 5-40
Lesson Summary 5-41
Lab: Configuring a Router 5-42
Basic router setup 5-42
The `ipforwarding` parameter 5-44
Relevant Commands and Files 5-46
Answers to Review Questions 5-47

Lesson 6: Configuring WAN Connections **6-1**

WAN Interface Types 6-6
Modems 6-6
ISDN 6-13
Other digital subscriber line technologies 6-14
Leased lines 6-17
Frame relay 6-19
Cable modems 6-20
WAN Protocols 6-21
Serial Line Internet Protocol (SLIP) 6-21
Point-to-Point Protocol (PPP) 6-23
PPP Configuration Overview 6-27
Configuring SCO PPP 6-29
Configuring the modem 6-29
Configuring UUCP 6-39
Configuring SCO PPP outgoing connections 6-43
Configuring SCO PPP incoming connections 6-56
Configuring SCO PPP dedicated connections 6-62
Advanced SCO PPP configuration 6-69
Configuring PPP from Morning Star 6-71
Outbound Morning Star PPP connections 6-73
Inbound Morning Star PPP connections 6-77
Advanced Morning Star PPP configuration 6-79
Troubleshooting PPP 6-80
Testing the connection with **cu**(C) 6-80

Using PPP Logs 6-82
Checking the PPP interface 6-83
Verifying network connections 6-83
Checking Routing 6-84
Reseting PPP 6-85
Lesson Summary 6-86
Lab: Configuring PPP Connections 6-88
Disabling LAN access 6-88
Configuring a modem for outgoing calls 6-88
Configuring */usr/lib/uucp/Systems* 6-89
Configuring an outgoing PPP connection 6-90
Configuring a modem for incoming calls 6-92
Configuring an incoming PPP connection 6-93
Relevant Commands and Files 6-95
Answers to Review Questions 6-97

Lesson 7: TCP/IP Administration Topics 7-1

TCP/IP Start-up 7-7
The */etc/tcp* file 7-7
Functions performed by */etc/tcp* 7-8
Maintenance-mode */etc/tcp* activities 7-11
Multiuser-mode */etc/tcp* activities 7-15
Network daemons not started 7-16
Starting and stopping TCP/IP 7-16
The **inetd**(ADMN) Super Daemon 7-19
How **inetd** works 7-19
Configuring **inetd** using the *inetd.conf* file 7-19
Enabling and disabling daemons 7-20
Using **telnet** to test daemons 7-21
Configuring Trusted Access 7-25
Configuring trusted host access 7-27
Configuring trusted user access 7-28
The trusted access sequence 7-35
Overview of user equivalence 7-39
Changing an account's UID and login name 7-44
Lesson Summary 7-45
Lab: Modifying TCP/IP startup 7-46
Relevant Commands and Files for TCP/IP Administration 7-48
Answers to Review Questions 7-50

Lesson 8: Domain Name System (DNS) Configuration 8-1

DNS Operation 8-9
DNS server types 8-11
How queries are answered 8-16
DNS Files 8-21

The *named.boot* file 8-23
The *named.hosts* file 8-37
The *named.rev* file 8-37
The *named.local* file 8-39
The *root.cache* file 8-39
The *named.soa* file 8-41
DNS Records 8-43
The `A` record 8-47
The `CNAME` record 8-47
The `PTR` Record 8-51
The `NS` record 8-53
The `SOA` record and `$include` 8-55
Configuring DNS servers 8-57
Configuring a primary name server 8-57
Configuring a secondary name server 8-59
Configuring a caching-only name server 8-60
Starting and Controlling the Name Server 8-61
Querying a name server 8-63
The **nslookup**(ADMN) command 8-63
The **dig**(ADMN) command 8-65
The **host**(ADMN) command 8-65
Configuring DNS Clients 8-67
The */etc/resolv.conf* file 8-67
Lesson Summary 8-72
Lab: Configuring and Testing Name Servers and Clients 8-74
Configuring servers 8-74
Testing your server 8-77
Configuring and testing a client 8-77
Relevant Commands and Files 8-79
Answers to Review Questions 8-81

Lesson 9: Configuring Internet Services **9-1**

Virtual Domains 9-7
Enabling virtual domains 9-7
Creating a virtual domain 9-11
Setting up virtual domain services 9-12
Web Services 9-14
Configuring the Netscape FastTrack Server 9-14
Running the Netscape Administration Server 9-17
Configuring remote administration 9-21
Configuring an FTP Server 9-27
Enabling FTP 9-27
Configuring anonymous FTP 9-29
Other FTP configurations 9-31
Flags for **ftpd**(ADMN) 9-40

Configuring Time Synchronization 9-41
Lesson Summary 9-44
Lab: Configuring your host to use NTP 9-45
Relevant Commands and Files 9-46
Answers to Review Questions 9-49

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

Firewall Basics 10-6
Packet filters 10-9
Static packet filters 10-9
Creating a PPP packet filter 10-13
Dynamic packet filters 10-17
Proxy servers 10-23
The Netscape Proxy Server 10-24
Using a proxy server 10-25
Packet Security 10-27
Packet sniffing 10-27
The Secure Socket Layer (SSL) 10-31
Lesson Summary 10-33
Lab: Implementing a PPP Packet Filter 10-34
Relevant Commands and Files 10-36
Answers to Review Questions 10-37

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

File and Printer Sharing Overview 11-7
NFS 11-7
VisionFS 11-8
Advanced File and Print Server 11-11
Installing and Using VisionFS 11-12
Installing VisionFS 11-12
Windows 95 network configuration 11-15
VisionFS administration overview 11-19
Sharing files and directories with VisionFS 11-23
Default settings and the master shared folder 11-25
Sharing printers with VisionFS 11-32
Remote Application Access 11-39
Terminal emulation 11-39
X Window servers 11-47
Custom client agents 11-48
Remote Boot Configuration 11-50
Configuring an Address Allocation Server 11-55
Configuring the DHCP server 11-57
Lesson Summary 11-60
Relevant Commands, Directories, Shares, and Packages 11-61
Answers to Review Questions 11-62

Appendix A: Mail Configuration **12-1**

Overview of e-mail 12-7
The Mail User Agent (MUA) 12-7
The Mail Transport Agent (MTA) 12-9
Mail delivery agent 12-9
POP and IMAP servers 12-10
Configuring sendmail 12-13
The default sendmail configuration 12-13
Modifying the sendmail configuration 12-14
Configuring mail aliases 12-21
User-configurable mail options 12-22
.forward files 12-23
Configuring DNS for use with mail 12-27
Enabling and Disabling Client-Server Mail (POP and IMAP) 12-30
MMDF 12-31
MMDF configuration 12-31
MMDF user-configurable features 12-32
Lesson Summary 12-34
Relevant Commands and Files 12-35
Answers to Review Questions 12-37

Appendix B: NFS **A-1**

NFS Concepts A-7
NFS terms A-7
Rules of importing and exporting A-11
Configuring Servers A-13
Using **scoadmin**(ADM) A-13
Configuring servers by hand A-18
NFS client configuration A-23
Using scoadmin(ADM) A-23
Configuring clients by hand A-26
NFS Startup and Shutdown A-32
The */etc/nfs* script A-32
NFS Security Issues A-34
Setuid and setgid bits on files on NFS servers A-34
Root privileges A-34
User equivalence revisited A-35
The NFS Automounter A-37
The automount process A-37
Configuring maps A-41
Configuring the automounter A-47
Lesson Summary A-49
Lab: NFS Administration A-51
Lab: Configuring the automounter A-53
Relevant Commands and Files A-55

Answers to Review Questions A-56

Lesson 3: Network Boot and Remote Configuration **B-1**

Overview of Booting from the Network B-6
The **RARP** method, **rarpd**(ADMN), and the */etc/ethers* file B-9
The BOOTP Protocol B-11
The */etc/bootptab* file B-13
Starting the **bootpd** daemon B-14
The DHCP Protocol B-15
Downloading the Device's Kernel B-17
The TFTP transfer method B-17
The NFS transfer method B-17
Security concerns B-17
Lesson Summary B-19
Relevant Commands and Files B-20
Answers to Review Questions B-21

1

Course Introduction

Instructor Guide

Lesson 1

Course Introduction

1.1 Introduction

This ***SCO OpenServer***™ Release 5 Network Administration course is designed for technically-oriented individuals who need to establish and maintain the network connectivity of their SCO OpenServer Release 5 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 SCO OpenServer 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
- 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 SCO OpenServer environment on a regular basis. Students are likely to have a wide variety of backgrounds in UNIX, SCO UnixWare, and SCO OpenServer. They must have prerequisite skills equal to or exceeding those provided in the ***SCO OpenServer Release 5 System Administration*** courses.

Course schedule

This course takes five days to complete. A suggested lesson duration is specified at the beginning of each lesson in this ***Instructor Guide***.

The labs included in the course must be completed in order to meet the course objectives.

Suggestions for practice exercises are also included throughout the course. Ample time has been allotted to each lesson to allow for the completion of these exercises.

Equipment

The following equipment should be installed in a classroom:

- One Pentium® or equivalent CPU computer per student in addition to the instructor's system. Each system should have:
  - A minimum of 500MB hard disk (1GB or larger recommended)
  - A minimum of 32MB of RAM (64MB is recommended)
- A minimum of one supported network adapter per system. In addition:
  - One system must have two network adapters. It is preferable that a minimum of two systems have two network adapters.
  - Network adapters with MDI drivers are strongly recommended (some labs, such as packet filters in Lesson 10, require MDI drivers).
- The classroom systems must be connected by a supported LAN technology, such as Ethernet or Token Ring, with enough cables, hubs, etc., to divide the classroom into at least two—preferably three or more—separate networks.
- A minimum of two supported modems per classroom; preferably one supported modem per system

- A minimum of two telephone lines or simulated telephone lines; preferably one line per modem
- An instructor's system configured to allow dial-in PPP connections
- A minimum of one networked Microsoft Windows 95 or NT system in the classroom, preferably one for every two students
- One liquid crystal display (LCD) unit, VGA projector, or large-screen display monitor
- White board with markers
- Overhead projector

Training materials

The following training materials are used in this course:

- ***Instructor Guide***
- ***Student Guide***
- Transparency masters
- vi Quick Reference Card
- ***Network Commands*** Job Aid
- Course graphics (transparencies) in GIF file format

In addition, the instructor should have the following printed material:

- ***SCO OpenServer Release Notes***
- ***SCO OpenServer Handbook***

Software

The following software is used in this course:

- The SCO OpenServer Release 5.0.5, Standard Enterprise System
- SCO PPP from Morning Star
- SCO VisionFS™ (Optional Services CD)

- SCO Internet Security Package (Optional Services CD)
- Netscape Proxy Server Package (Optional Services CD)

The *Instructor Guide*

The ***Instructor Guide*** (***IG***) is organized to include:

- Instructor notes are intermingled with text in the ***Student Guide*** (***SG***) allowing the instructor to easily refer to section content.
- Page numbers in the ***IG*** do not correspond to those in the ***SG***, however the heading numbers do.
- Cross references to other sections within the course materials are included within the text. These cross references generally include the section number or AV number, the section name, and page number. The page number is almost always different in the ***SG*** than the ***IG***; however, the other components are the same.

The ***IG*** also contains notes which are instructional aids, including:

AV 1-1: Slide Title
These notes explain the instructional objective of every transparency. The AV number cross references to the number that appears above every slide in the workbook.

IG Notes provide extra information not presented in the ***Student Guide*** intended for more advanced students.

1.2 Student Materials

The contents of the box containing your student materials are yours to keep and write notes in. Your instructor may 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 are 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 may refer to the following SCO printed documentation:

SCO OpenServer Release Notes	Contains important late-breaking information about installations, hardware requirements, and known limitations
SCO OpenServer Handbook	Provides the information you need to get your SCO OpenServer Release 5 system up and running. It also covers many common 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**(XC).

Lesson plan

Each lesson follows a similar approach, or methodology, in the delivery of instruction:

- The suggested lesson duration and instructor preparation notes are provided at the beginning of each lesson, when applicable.
- The lesson begins with an overview transparency of the major topics and a list of the lesson objectives.

- The lesson continues with the presentation of lesson topics and the demonstration of lesson tasks.
- Exercises, which allow the students to practice discrete tasks, appear throughout the presentation.
- References appear where appropriate.
- The lesson content concludes with a summary. Use this to reinforce the topics and tasks covered. You should also refer to the objectives that were presented at the beginning of the lesson.
- Following the content of most lessons are summary exercises, which consist of review questions and labs. The review questions allow the instructor to determine how well the students understood the content, and the labs allow students to apply the material that they learned.

1.3 Typographical Conventions

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

lp or **lp**(C)	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
/new/client.list	Files, directories, desktops (names, icons, or windows), system, network, and printer names
root	User names
arg or *`arg`*	Placeholders (replace with appropriate name or value)
`[ arg ]`	Surrounds an *option* or *cmdarg* that is not mandatory
`arg` ...	Indicates multiple occurrences of the *option* or *cmdarg*
$	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
F2	Keyboard keys
CTRL SPACE	Press CTRL and SPACE keys simultaneously
CTRL ALT DEL	Press all three keys, CTRL, ALT, and DEL, simultaneously
CTRL d	Press CTRL key. With CTRL still depressed, press D.
ALT 0240	Press ALT key. With ALT still depressed, press and release in sequence 0, 2, 4, and 0.
CTRL Q + i	Press CTRL key. With CTRL still depressed, press Q. Release both keys and press I.
CTRL Q + I	Press CTRL key. With CTRL still depressed, press Q. Release both keys and press SHIFT I.
`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**(S)	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
employees	Database names and tables
`buf`	C program structures
`b_b.errno`	Structure members

1.4 Using scoadmin(ADM)

Many procedures in this course indicate to "start the **Network Configuration Manager**" or "run **scoadmin**(ADM)" when invoking administration utilities.

SCO OpenServer Release 5 includes both graphical and character-based versions of most administration utilities. When started from the Desktop, the graphical versions automatically run. When started from a character device (such as a terminal), the character versions run.

In this course, the Desktop is used to start the graphical versions of administration utilities. There are several ways to run **scoadmin**(ADM) utilities.

For example, to run the **Network Configuration Manager,** a tool that allows you to configure network adapter drivers and manage network connections:

- From the **SCOadmin** icon, select **Networks ⇨ Network Configuration Manager**.
- From the command line, enter **`netconfig`**.
- From the command line, enter **`scoadmin network config`**.

When you are directed to "run **netconfig**(ADM)" or "start the **Network Configuration Manager**" later in this guide, you can use any of these methods.

Be sure you are logged in as **root**.

Classroom setup

Install the necessary software:

- During operating system installation, select **sendmail** as the mail transfer agent. Note that MMDF is the default.
- Install the class software on all the machines.

Note that later in the course, the students will remove SCO PPP and install Morning Star PPP. They will also install VisionFS from the Optional Services CD.

Configure classroom equipment, your own display, and the LCD.

Lesson preparation

- Read the pre-lesson preparation prior to each lesson. Complete any tasks listed in the preparation.
- Read the lesson thoroughly before teaching, and add any additional notes or exercises.
- Read through the appropriate sections of the SCO documentation in support of the lesson topics.
- Review the system files in each lesson to which students need access, and change the file permissions to make the files readable if necessary.

1.5 Sample exam items

The following 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. The following text is found in the file */etc/named.boot*:

```
directory  /etc/named.d

primary    chicago.us.org           named.hosts
primary    75.43.197.in-addr.arpa   named.rev
primary    0.0.127.in-addr.arpa     named.local
secondary  can.org                  1.2.3.4   can.bak
secondary  22.97.201.in-addr.arpa   1.2.3.4   can.rev
```

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

A. `host1  IN  A  197.43.75.1`

B. `host1  IN  MX  10  mailgate.us.org.`

C. `1  IN  PTR  host1.us.org.`

D. `ftp.us.org  IN  CNAME  host1`

3. Which of the following is **NOT** supported by VisionFS?

A. Sharing UnixWare files to Windows systems

B. Sharing Windows files to UnixWare systems

C. Sharing UnixWare printers to Windows systems

D. Sharing Windows printers to UnixWare systems

Answers to sample exam items

1. B
2. C
3. B

2

TCP/IP Concepts

Instructor Guide

Lesson 2

TCP/IP Concepts

Suggested lesson duration

This lesson should take approximately 4 hours to complete. There is no lab exercise.

Pre-lesson classroom preparation

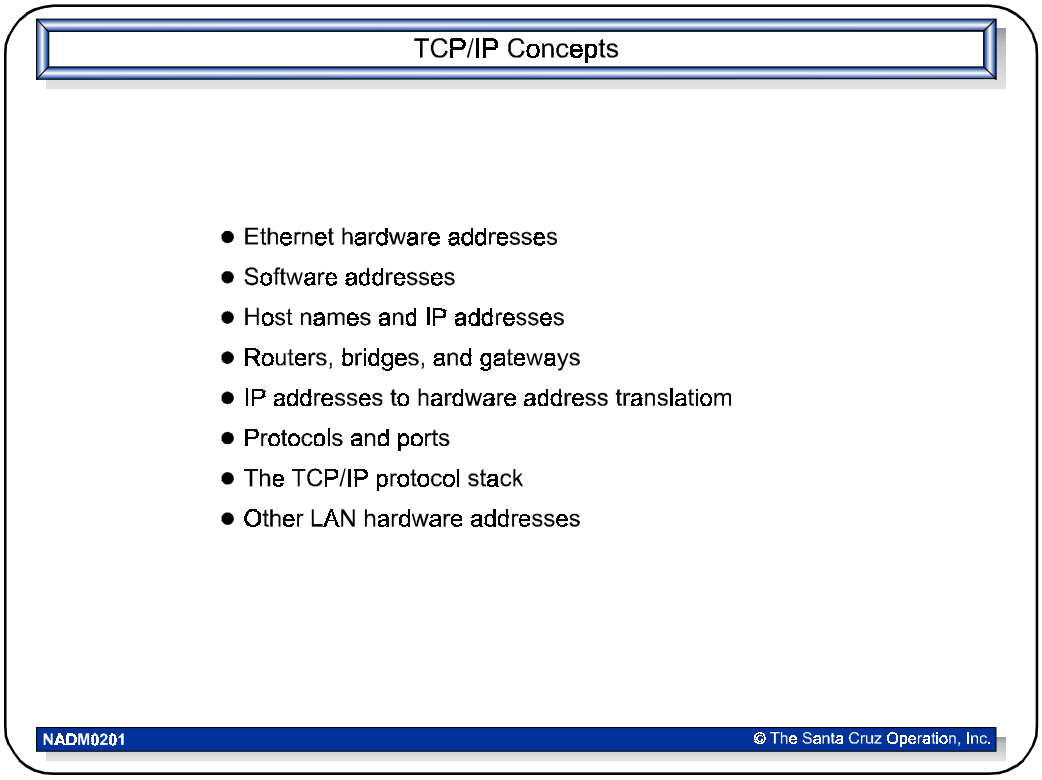
- Install the network adapter cards prior to the start of class.
- Install and configure TCP/IP on the instructor's machine prior to the start of class.

Instructor preparation

The following textbooks may be useful for preparing for this lesson:

- Comer, Douglas E.***, Internetworking with TCP/IP, Volume I: Principles, Protocols, and Architecture***, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1995, ISBN: 0-13-216987-8
- Loshin, Peter, ***TCP/IP for Everyone***, Chestnut Hill, MA: Academic Press, 1995, ISBN: 0-12-455827-5
- Stevens, W. Richard, ***TCP/IP Illustrated, Volume 1: The Protocols***, Reading, MA: Addison-Wesley, 1994, ISBN: 0-201-63346-9
- Ford, Merilee, Lwe, H. Kim, Spanier, Steve, and Stevenson, Tim, ***Internetworking Technologies Handbook***, Indianapolis, IN: New Riders Publishing, ISBN: 1-56205-603-4
- Nassar, J. Daniel, ***Ethernet and Token Ring Optimization***, New York, NY: M & T Books, ISBN: 1-55851-464-3
- Hunt, Craig, ***TCP/IP Network Administration***, Sebastopol, CA, O'Reiley & Associates, Inc., ISBN: 0-937175-82-X

AV 2-1


TCP/IP Concepts
Ethernet hardware addresses
Software addresses
Host names and IP addresses
Routers, bridges, and gateways
IP addresses to hardware address translatiom
Protocols and ports
The TCP/IP protocol stack
Other LAN hardware addresses
NADM0201
© The Santa Cruz Operation, Inc.

Introduction

The SCO OpenServer 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 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

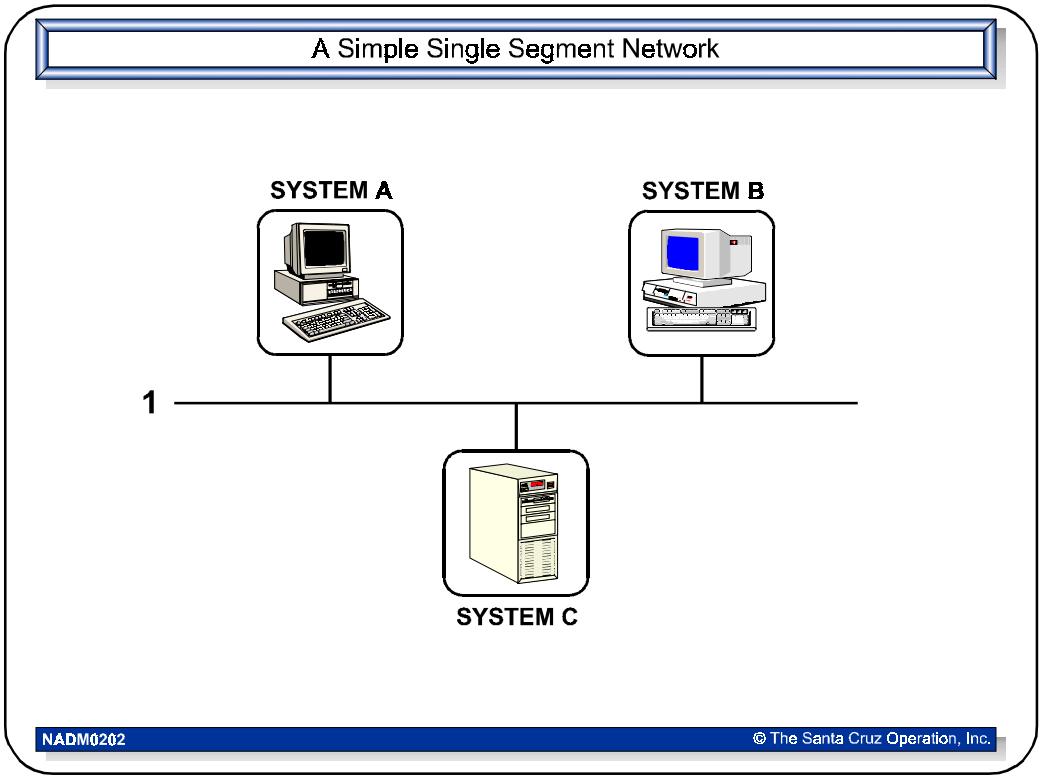
This lesson describes the basic components of a computer network. By the end of this lesson, you will be able to:

- 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/services* and */etc/hosts* files
- List the types of network adapters supported by SCO OpenServer

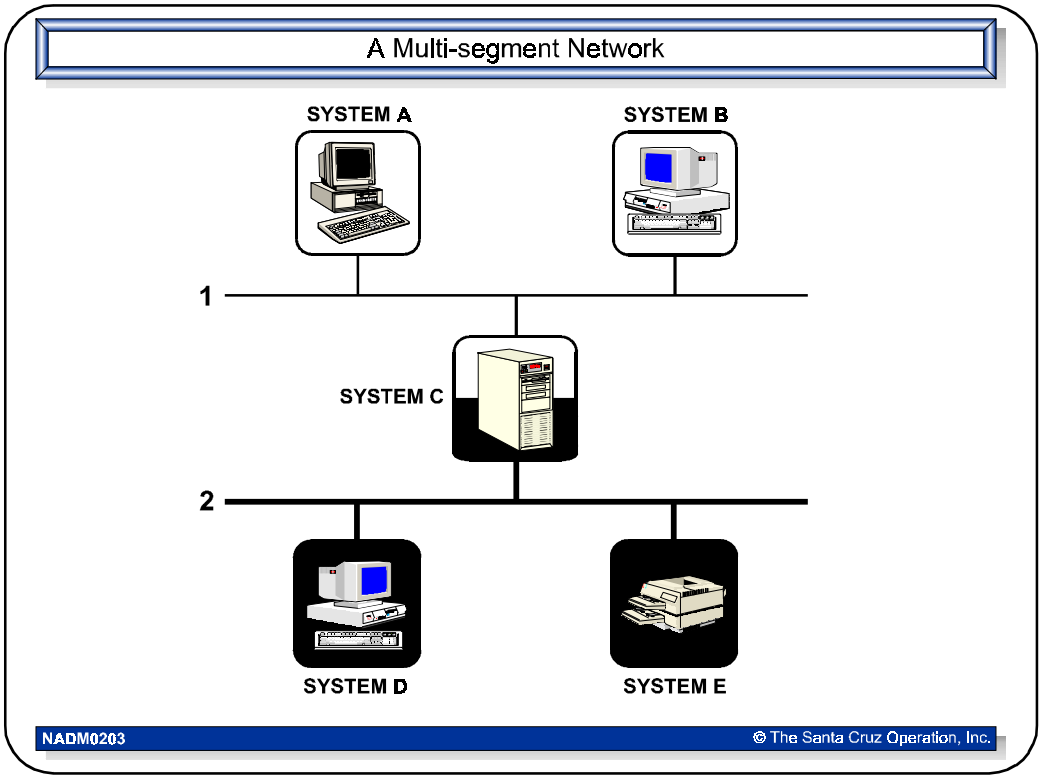
This course uses the term network adapter instead of network interface or NIC.

AV 2-1: TCP/IP Concepts
The first slide describes the overall topics covered in this lesson

AV 2-2


A Simple Single Segment Network
SYSTEM A
SYSTEM B
1
SYSTEM C
NADM0202
© The Santa Cruz Operation, Inc.

AV 2-3


A Multi-segment Network
SYSTEM A
SYSTEM B
1
SYSTEM C
2
SYSTEM D
SYSTEM E
NADM0203
© The Santa Cruz Operation, Inc.

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

The term network adapter is used in place of NIC to avoid confusion with the term InterNIC.

  - *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
      ```

Each hardware vendor is assigned its own address range, so it is possible to identify the maker of a network adapter by its address. Some vendors allow users to change the non-vendor portion of the address. In this case, take care that no two network adapters on a given subnet have the same hardware address.

  - *Token Ring* addresses are six fields of hexadecimal numbers. No two adapters on a given ring can have the same address. An example Token Ring address is:

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

Each message sent using the network uses two unique addresses:

- A *destination address* identifies which system is to receive the message.
- A *source address* identifies which system sent the message.

A simple analogy is mail sent through the U.S. Postal Service (USPS). A unique destination address indicates where the mail is to be delivered. The source (return) address indicates where it came from so the addressee can reply if necessary. The USPS can also use the return address to notify the sender in the event there are delivery problems.

AV 2-2: A Simple Single Segment Network
This slide shows a simple network ("Network #1") with three nodes, *A*, *B*, and *C*. Use this slide to indicate that the systems can easily communicate with each other by indicating the address of the desired recipient (the destination address). Replies are facilitated with the source address.

Note that if *A* wants to talk to *B*, it does so directly. System *C* is not involved even though the picture might indicate that messages would pass through it first.

Instructor Guide

AV 2-3: A Multi-segment Network
This slide shows two different "simple single-segment networks." Network #1 is made up of *A*, *B*, and *C*; Network #2 with *C*, *D*, and *E*. (Mention that System *C* has two network adapters, so it can be on two different networks.)

Ask the students about sending messages between the machines on a given network, such as *A* to *B* or *C* to *E*. There is no problem because the machines are connected and have the needed address.

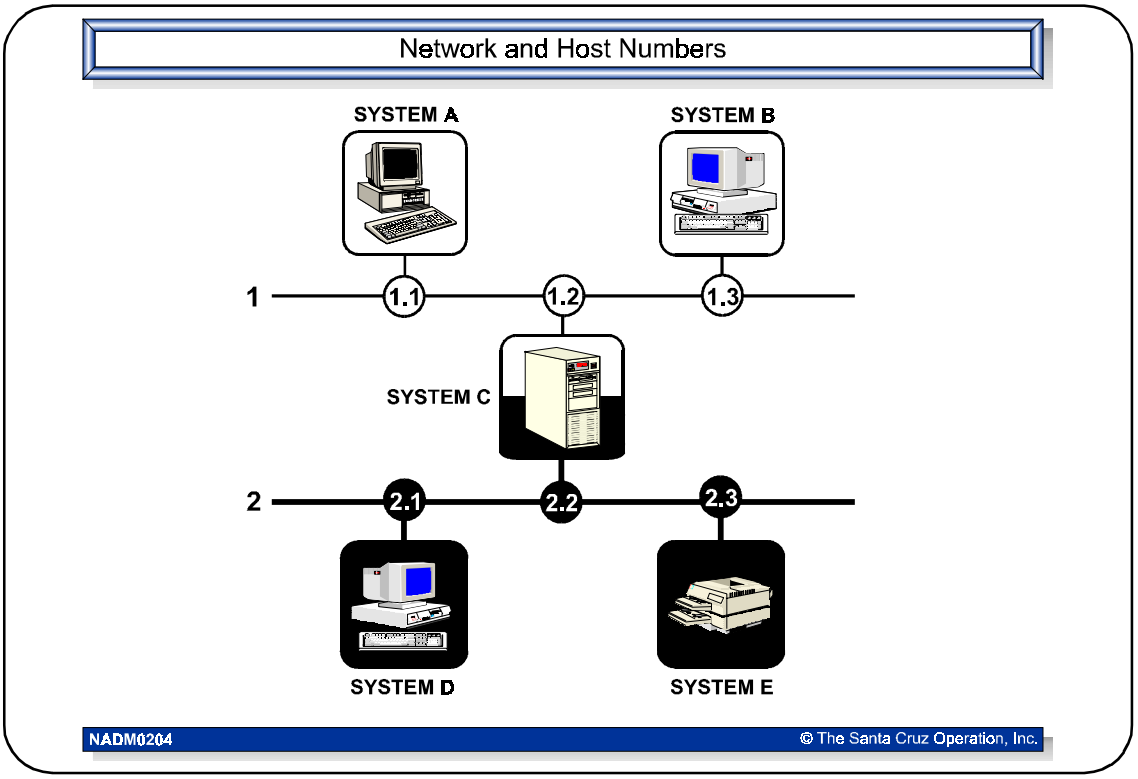
Ask them if System *A* can send a message to System *E*. It can do so by going through System *C*. System A cannot talk to *E* directly because they are on different networks. System *C*, however, is on *both* networks, so it can "route" the message along.

As a segue to software addresses, start a discussion about the following issues:

- What information is there in the Ethernet hardware address that indicates that the message needs to be routed using *C*? The answer is nothing!
- Assume that the systems keep track of other systems' hardware addresses with some kind of table. What would happen if the network adapter in one machine is replaced with another network adapter (which, of course, has a different hardware address). How would you go about informing the other systems about a change of address?

These issues are why we have software addresses in addition to hardware addresses, which is discussed next.

AV 2-4


Network and Host Numbers
SYSTEM A
SYSTEM B
1
1.1
1.2
1.3
SYSTEM C
2
2.1
2.2
2.3
SYSTEM D
SYSTEM E
NADM0204
© The Santa Cruz Operation, Inc.

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

The term *interface* is probably a better term to use than *host* because the device being addressed could be a printer, terminal server, X-terminal, etc, as well as a "host" computer. "Host" is still commonly used in SCO documentation and other network textbooks, so it is used here as well to avoid confusion. In addition, one host may contain multiple interfaces, in which case each interface is assigned its own IP address. This is illustrated by System C in the slide on AV 2-4.

AV 2-4: Network and Host Numbers
This slide shows the network and host numbers for the nodes in the two networks. Note that this is a simplified numbering scheme with only one number per component (unlike real IP addresses which can have multiple numbers for a given component). Point out the dot which is used as a separating character.

Note that System *C* has two addresses: 1.2 and 2.2. This is because it has two network adapters, one for each network. Coincidentally, the host portion for System *C* was the same on both Network 1 and 2; this is not a requirement.

2.2.1.1 IP address syntax

The network number must be unique only if it is going to communicate with other *established* networks (e.g., the Internet). The host numbers are unique on a *given* network, but a given host number can be used on a different network as well.

As an analogy, telephone numbers use an area code (network number) and a local number (host number). Local numbers can be duplicated in different area codes, but not within a given area code. The area codes themselves are not duplicated because they need to communicate with other established "networks" (i.e., area codes).

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:

The term *octet* is preferred to the term *byte*. An octet is an 8-bit value; the size of a byte is machine-dependent.

```
132.147.210.9
```

This example address is the IP address of *websco.sco.com*, the SCO anonymous ftp server.

- Each octet is an 8-bit value and is limited to a decimal range of 0–255.

Although the possible range is 256 numbers, some numbers in some positions are reserved, so the actual number of usable values is lower.

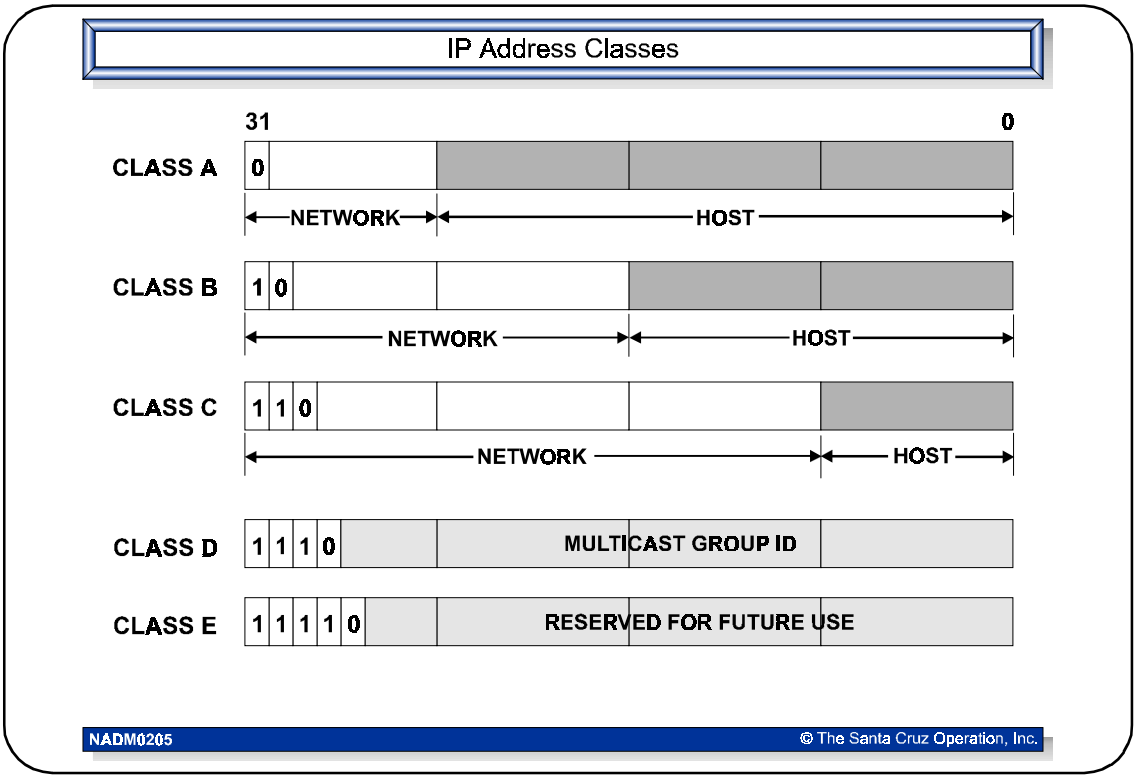
- One or more octets comprise the network portion of the IP address; one or more octets comprise the host portion. There are four octets in total.

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 can exist on the entire network? How did you calculate this number?

The answer is 4294967296 (2^{32}).

AV 2-5


IP Address Classes
31
0
CLASS A
0
NETWORK
HOST
CLASS B
1 0
NETWORK
HOST
CLASS C
1 1 0
NETWORK
HOST
CLASS D
1 1 1 0
MULTICAST GROUP ID
CLASS E
1 1 1 1 0
RESERVED FOR FUTURE USE
NADM0205
© The Santa Cruz Operation, Inc.

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 of the bits are reserved for the network portion, then you can have many different networks, but relatively few hosts on each network.
- If too many of the 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 most widely used.

Class D addresses are used for multicast addresses.

Class E addresses are "reserved for future use," but are used for experimental purposes today.

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

AV 2-5: IP Address Classes
This slide shows the allocation of the four octets (bits 0–31) in an IP address for the different classes.

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

- Class A 1–127
- Class B 128–191
- Class C 192–223
- Class D 224–239
- Class E 240–247

Note that 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. Therefore:

- The number of class A networks is $2^7 - 2 = 126$. They range from 1 to 126.
- The number of class B networks is $2^{14} - 2 = 16{,}382$. They range from 128.1 to 191.254.

- The number of class C networks is 2^{21} - 2 = 2,097,150. They range from 192.0.1 to 223.255.254.

Review the math involved in calculating the range of numbers for each class:

- Class A: The first bit is 0. If the remaining 7 bits are 0, you get 0000 0000 (i.e., 0x00), which is 0, for the low range. If the remaining 7 bits are 1, you get 0111 1111 (i.e., 0x7F), which is 127, for the high range.
- Class B: The first two bits are 10. If the remaining 6 bits are 0, you get 1000 0000 (i.e., 0x80), which is 128, for the low range. If the remaining 6 bits are 1, you get 1011 1111 (i.e., 0xBF), which is 191, for the high range.
- Class C: The first three bits are 110. If the remaining 5 bits are 0, you get 1100 0000 (i.e., 0xC0), which is 192, for the low range. If the remaining 5 bits are 1, you get 1101 1111 (i.e., 0xDF), which is 223, for the high range.
- Class D: The first four bits are 1110. If the remaining 4 bits are 0, you get 1110 0000 (i.e., 0xE0), which is 224, for the low range. If the remaining 4 bits are 1, you get 1110 1111 (i.e., 0xEF), which is 239, for the high range.
- Class E: The first five bits are 11110. If the remaining 3 bits are 0, you get 1111 0000 (i.e., 0xF0), which is 240, for the low range. If the remaining 3 bits are 1, you get 1111 0111 (i.e., 0xF7), which is 247, for the high range.

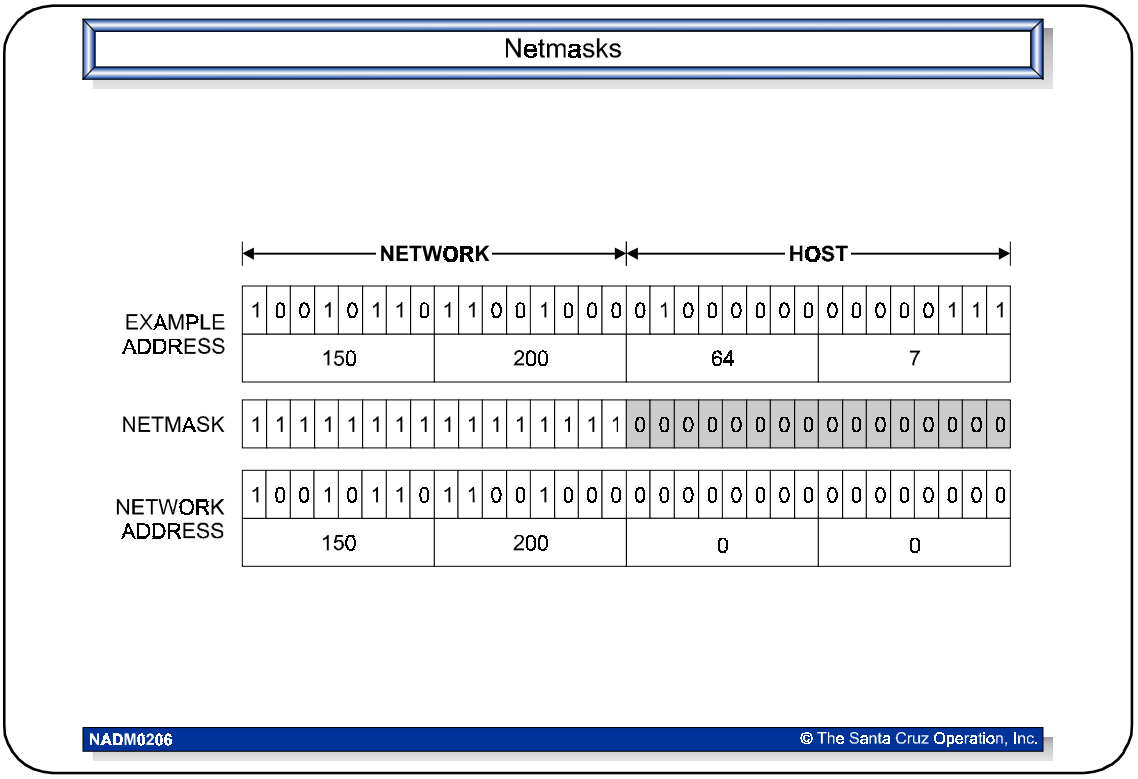
Instructor Guide

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

Messages destined to a single host (the most common scenario) use a unicast address. Broadcasting and multicasting only apply to UDP, where it makes sense for an application to send a single message to multiple recipients. TCP is a connect-oriented protocol that implies a connection between two hosts (specified by IP addresses) and one process on each host (specified by port numbers).

AV 2-6



Instructor Guide

2.2.2 Netmasks

A *netmask* indicates how many bits are used for the network address and how many for the host address. More specifically, it is used to determine 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.

AV 2-6: Netmasks

This slide shows how netmask values are generated. The Class B network described above is used as the example. Point out that this example *must* be a Class B address because the first two bits in the network address are 10.

Go over the example address and how the netmask masks off the network portion of that address.

Exercise: Specifying netmasks

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

Assume these networks do not use subnetting.

- Class A network
- Class C network

The answers are:

- Class A network: 255.0.0.0
- Class C network: 255.255.255.0

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, "The routing table," on page 5-17).

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 host 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 detail in Lesson 4: "Subnets."

2.2.4 Broadcast addresses

This section assumes addresses that do not use subnetting. Broadcasting with subnet addresses is revisited in Lesson 4: "Subnets."

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 acknowledgment of receipt of the packet.

It is possible for the receivers to send their own messages back to the sender acknowledging receipt of the packet. However, this feature is not built into the broadcast mechanism.

- Addressed to all hosts on a network. Therefore, the packets are not normally forwarded by a router to other networks.

Examples of programs using broadcast packets include:

Routing daemons	Announces routes they are aware of to other machines running routing programs.
rwhod(ADMN)	Announces usage metrics on the sending machine to other machines running **rwhod**(ADMN). These statistics can be seen with the **rwho**(TC) and **ruptime**(TC) commands.
ping(ADMN)	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 150.200.76.17
- Netmask of 255.255.255.0

The broadcast address of the interface is 150.200.76.255.

Make sure students understand this. Provide other examples to make sure students have a firm understanding of broadcast addresses.

Exercise: Determining broadcast addresses

NOTE: This exercise assumes networks that are not subnetted.

Assuming no subnetting, given the following IP addresses:

- 85.110.43.7
- 210.20.210.20
- 150.75.37.5

Determine the:

- Netmask
- Network address
- Broadcast address

The answers are:

- 85.110.43.7 is a class A address that yields a:
  - 255.0.0.0 netmask
  - 85.0.0.0 network address
  - 85.255.255.255 broadcast address
- 210.20.210.20 is a class C address that yields a:
  - 255.255.255.0 netmask
  - 210.20.210.0 network address
  - 210.20.210.255 broadcast
- 150.75.37.5 is a class B address that yields a:
  - 255.255.0.0 netmask
  - 150.75.0.0 network address

- 150.75.255.255 broadcast address

2.2.5 Available host addresses

Given the discussion of address classes, netmasks, network addresses, and broadcast addresses, you can now determine the number of hosts that can exist on a network. This number is necessary to plan the layout of your network.

The number of hosts that can exist on a network:

- Depends upon the class of the address and its netmask
- Is given by the 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
  - Class B address can have $2^{16} - 2$ (65534) hosts
  - 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 2-20) cannot be assigned to a host. (For example, the host portion of an IP address cannot contain all zeros).

For further information on why the host portion cannot be 0, see RFC 950.

  - One of the host addresses (normally all ones) must be used as the network’s broadcast address.

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 multiple 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 that each year, during SCO Forum, the 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

Multicasting works in the following way:

1. A multicasting program selects a multicast address on which to send its data. These addresses are allocated by a Network Information Center (NIC) just like other IP addresses.
2. The packets are broadcast on the LAN of the sender machine (unless multicasting is supported by the hardware in use on the LAN).
3. Routers on the LAN notice that the destination IP address is a multicast address and re-broadcast the packet to the other LANs to which they are attached.
4. Programs, having registered themselves as willing to receive multicast packets at the indicated address, receive the packets and (in this example) play them on the local machine's sound card.

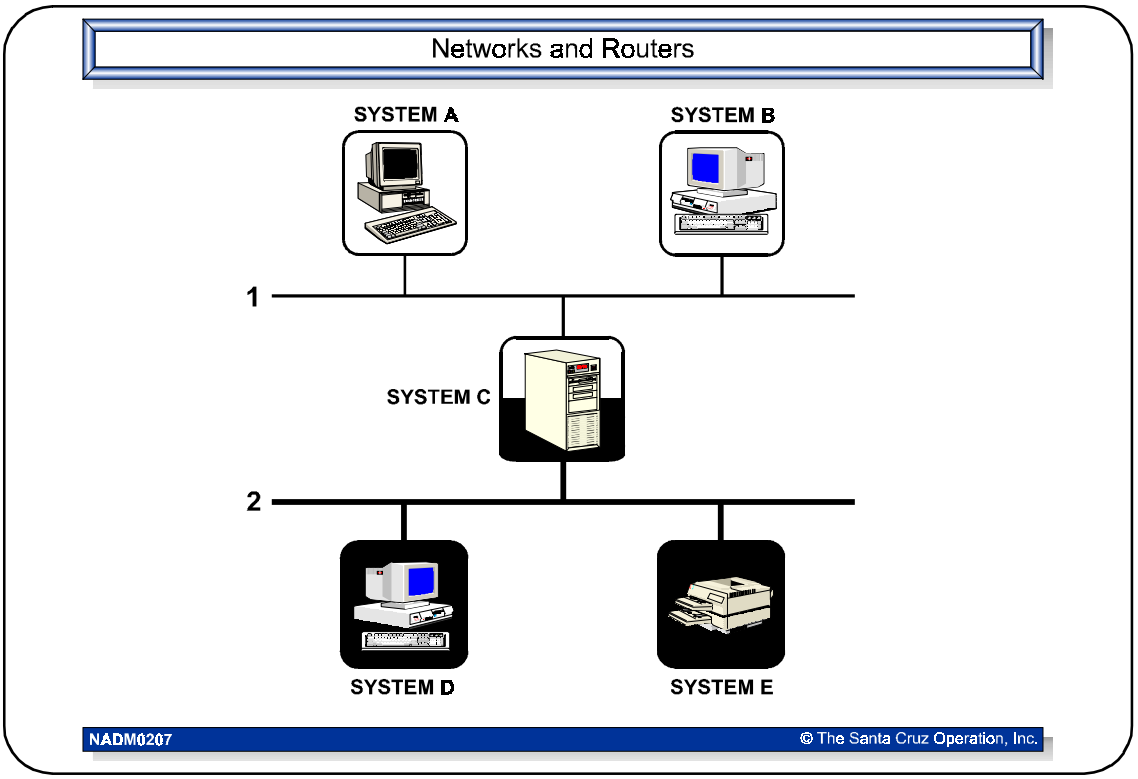
Instructor Guide

The registration is done using the Internet Group Management Protocol (IGMP). This allows multicast routing daemons to maintain a list of hosts (on their networks) that are listening on a particular multicast address. If no hosts are listening, the router does not have to re-broadcast the multicast packets.

SCO OpenServer systems can function as clients and receive multicast packets. However, the routing protocols in SCO OpenServer Release 5 do not support multicasting.

Note that UnixWare 7 can be a multicast router, using **mrouted**.

AV 2-7


Networks and Routers
SYSTEM A
SYSTEM B
1
SYSTEM C
2
SYSTEM D
SYSTEM E
NADM0207
© The Santa Cruz Operation, Inc.

2.3 Network-Joining Devices

AV 2-7: Networks and Routers
Use this slide to show the general nature of how a router fits into a network.

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. Some reasons include:

- 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

To extend your network, there are three options:

- 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.

Repeaters are not as widely used because hubs perform this functionality more effectively.

  - 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.

Hub technology is popular for Ethernet networks. You might need to explain the different wiring schemes (i.e., thick-wire Ethernet (thicknet), thin-wire Ethernet (thinnet), and twisted pair (10BASE-T) wiring). These are discussed in more detail in Section 2.8.1, “Ethernet,” on page 2-54.

- 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.

Because this avoids collisions, backoffs, and so on, if you are using a 100Mbps network technology, each transfer can theoretically be at 100Mbps (although in reality, few switches scale bandwidth linearly, due to expense).

Because traffic is isolated to only the sender and receiver by the switch, security is enhanced because packet snooping is almost impossible.

- 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.

Routers can only direct traffic between networks that use the same protocol. Gateways commonly appear at the application layer in the protocol stack; routers are at the IP layer.

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 more easily remember and use *names* than IP *addresses*. Some software is designed to work only with 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 using the *resolver.*
- The resolver traditionally uses the */etc/hosts* file.

Host name to IP address mapping can also be done using **named**(ADMN) or NIS.

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 the following properties:

- Consist of lowercase letters, numbers, and dashes
- Begin with a letter
- Contain no more than 8 characters

Some older systems may 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:

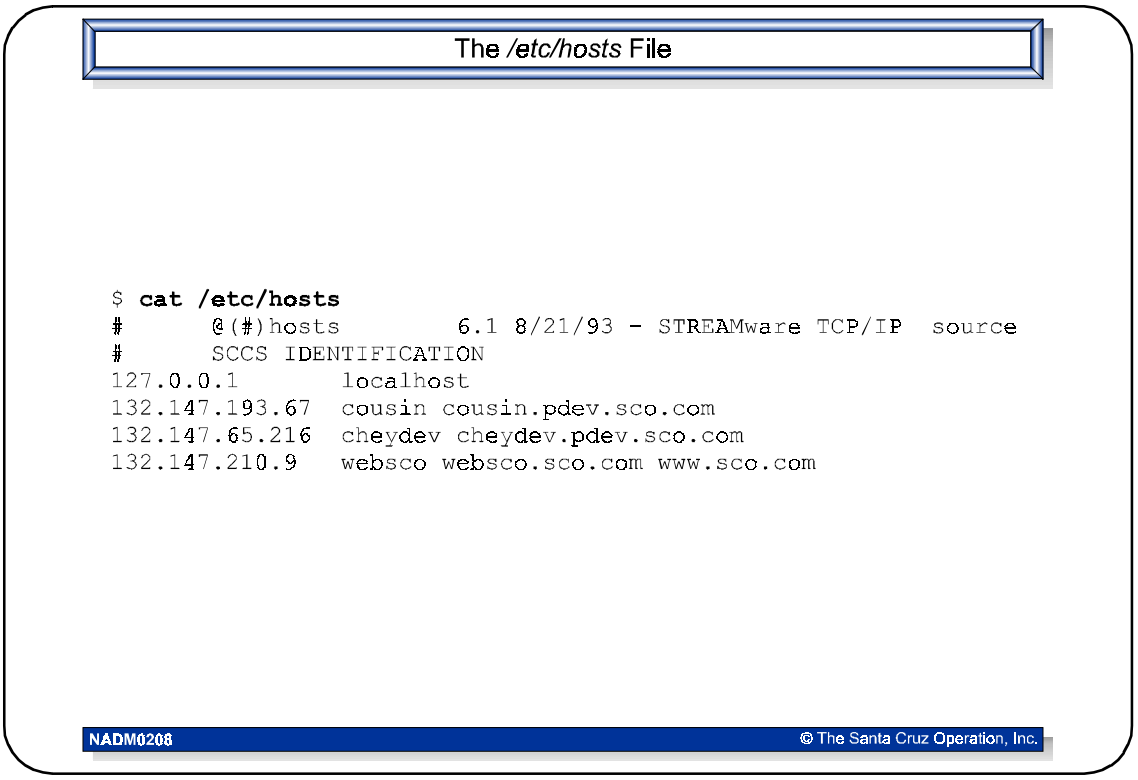
- All machines belonging to the United Nations are in the *un.org* sub-domain.
- All machines belonging to SCO are in the *sco.com* sub-domain.

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. Normally, 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 cousin.pdev.sco.com
132.147.65.216	cheydev cheydev.pdev.sco.com
132.147.210.9	websco websco.sco.com www.sco.com
NADM0208
© The Santa Cruz Operation, Inc.

2.4.3 The */etc/hosts* file

AV 2-8: The */etc/hosts* File
This slide shows a sample */etc/hosts* file. Point out that it is an ASCII text file that can be edited with any text editor (such as **vi**(C)). Note the use of the alias in the last example and tell students that something like this is often done for web servers (though normally with DNS instead of */etc/hosts*).

The */etc/hosts* file (see *hosts*(SFF)) 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 the node name of the host. This is the same as the output from the command:

  uname -n

- The *aliasname* is a list of alternate names for the host. It is suggested that you specify the fully qualified domain name as one of the aliases.

  NOTE: If using MMDF, the fully qualified name must come before the simple host name (alias). Alternately, you can make separate entries for each on individual lines in the file. An IP address can appear more than once.

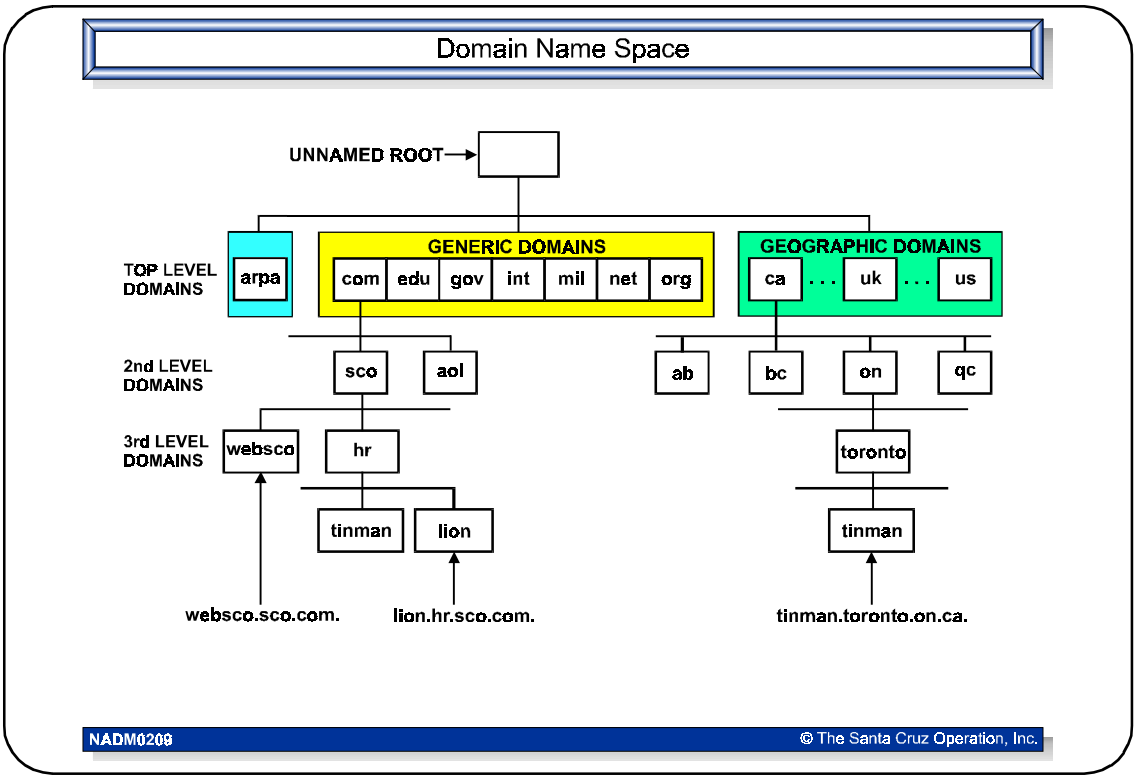
- The routines that use this file ignore comments (portions of lines beginning with a pound (#) character) and blank lines.

- 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. This line in the */etc/hosts* file should not be modified.

Exercise: Configuring */etc/hosts*

Edit the */etc/hosts* file on your system and add all the IP addresses and host names of the systems used in the classroom. Your instructor will provide the IP addresses and host name information.

AV 2-9


Domain Name Space
UNNAMED ROOT
TOP LEVEL DOMAINS
arpa
GENERIC DOMAINS
com
edu
gov
int
mil
net
org
GEOGRAPHIC DOMAINS
ca
uk
us
2nd LEVEL DOMAINS
sco
aol
ab
bc
on
qc
3rd LEVEL DOMAINS
websco
hr
toronto
tinman
lion
tinman
websco.sco.com.
lion.hr.sco.com.
tinman.toronto.on.ca.
NADM0209
© The Santa Cruz Operation, Inc.

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. See "Configuring the Network Information Service (NIS)" in the ***SCO OpenServer Networking Guide***.

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

2.4.3.1 Domain name space

AV 2-9: Domain Name Space
This slide shows the domain name space hierarchy. Point out that all the generic and geographic domains also have similar branches under them—they are omitted in the diagram for clarity.

Section 2.4.2, "Domain names," on page 2-29 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.

Note that the root domain has no "name," but it does have a path (namely, "dot" (**.**)). This is similar to the UNIX filesystem whose root directory has no name, but has the path "slash" (**/**).

- 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 8.3, "DNS Records," on page 8-43.)

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*)

The net domain is generally used for Internet infrastructure organizations.

- Other non-profit organizations (*org*)

- Two-character geographic domain names, based on the country codes found in ISO 3166. Some examples include:
  - The United Kingdom (*uk*), Germany (*de*), France (*fr*)
  - Canada (*ca*), the United States (*us*)

There are 54 subdomains for the states and territories of the USA, for example California (*ca*), Maryland (*md*).

NOTE: The InterNIC in the United States is reorganizing the domain names. This course was current as of its writing. Check *http://www.internic.net* for the latest organization of domains in the United States.

Instructor Guide

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).

For more information, refer to **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Domain Name Service ⇨ DNS domains and zones ⇨ 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.

When a user connects to a host without specifying a fully qualified name (e.g., *lion*), and that host is in the same domain as the user, DNS software appends the domain name (e.g., *hr.sco.com*), saving the user some typing.

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

There is room for confusion here. A NIS domain name is not the same as a DNS domain name. However, NIS files often contain data that includes DNS domain names.

- 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).

Exercise: Locating all references to the domain name

The object of this exercise is to find all of the files in the root filesystem that contain your domain name.

Step 1. Log in as **root.**

Step 2. Enter this command:

```
find / -type f -exec \
fgrep -l 'domain' {} \; > /tmp/domainname
```

where ***domain*** is the domain name of your machine.

It is a good idea to run this as a background job.

Step 3. When finished, view the */tmp/domainname* file and discuss the results with your instructor.

2.5 Translating IP Addresses to Hardware Addresses

So far, you have seen 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 requesting machine.

2.5.2 The ARP cache and the arp(ADMN) 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 can 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 */etc/default/inet* file contains parameters that control the behavior of the Address Resolution Protocol. These include:

`arpprintfs`	Controls logging of warnings from the kernel ARP driver. These are displayed on the console. If set to 0 (the default), debugging information is not displayed.

`arp_maxretries`	Sets the maximum number of retries for the address resolution protocol (ARP) before it gives up. The default value is 5; the minimum and maximum configurable values are 1 and 128.
`arpt_down`	Sets the time to hold onto an incomplete ARP cache entry if ARP lookup fails. The default value is 20 seconds; the minimum and maximum configurable values are 1 and 600 seconds.
`arpt_keep`	Sets the time to keep a valid entry in the ARP cache. The default value is 1200 seconds; the minimum and maximum configurable values are 1 and 2400 seconds.
`arpt_prune`	Sets the interval between scanning the ARP table for stale entries. The default value is 300 seconds; the minimum and maximum configurable values are 1 and 1800 seconds.

NOTE: These parameters should only be modified using the **inconfig**(ADMN) command.

The arp cache can be viewed with the **arp**(ADMN) 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 (150.200.75.18) at 8:0:9:69:ac:46
```

If the user's PATH variable includes the */etc* directory as a search component, it is not necessary to specify the full pathname of the command on the command line. The **root** user's PATH normally has */etc* included.

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

```
$ arp -a
mother.dev.sco.COM (150.200.75.18) at 8:0:9:69:ac:46
father.dev.sco.COM (150.200.75.19) at 0:80:5f:20:e7:d0
teacher.dev.sco.COM (150.200.75.190) at 0:0:c0:92:4b:24
```

Refer to **arp**(ADMN) 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 time-of-day server returns the current time, typically in a single datagram. A file server, however, commonly receives and sends multiple datagrams.

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 occur on a single machine. Network applications are developed using the IPC model.

IPC mechanisms include pipes, semaphores, and shared memory. Students should already have an understanding of the way pipes allow one process to communicate with another.

It is usually easier to develop client software than server software.

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 this 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/services* file.

Refer to *services*(SFF) for more information on */etc/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.

If the new port value is greater than 1024, you run the risk of ordinary users pretending to be the indicated server software and compromising system integrity. This is an important security issue that you might want to cover in greater detail, depending upon your audience.

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.

"Connection-oriented" means that before transmitting data, participants must establish a connection.

- 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**(ADMP) and **udp**(ADMP) for more information on these protocols.

Exercise: Identifying ports and protocols

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

- **netstat**

netstat uses port 15, on TCP.

- **ftp**

ftp uses 21 on TCP. FTP also makes use of port 20 on TCP (the ftp-data service).

- **who**

who uses port 513 on UDP.

- **login**

login uses port 513 on TCP.

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.

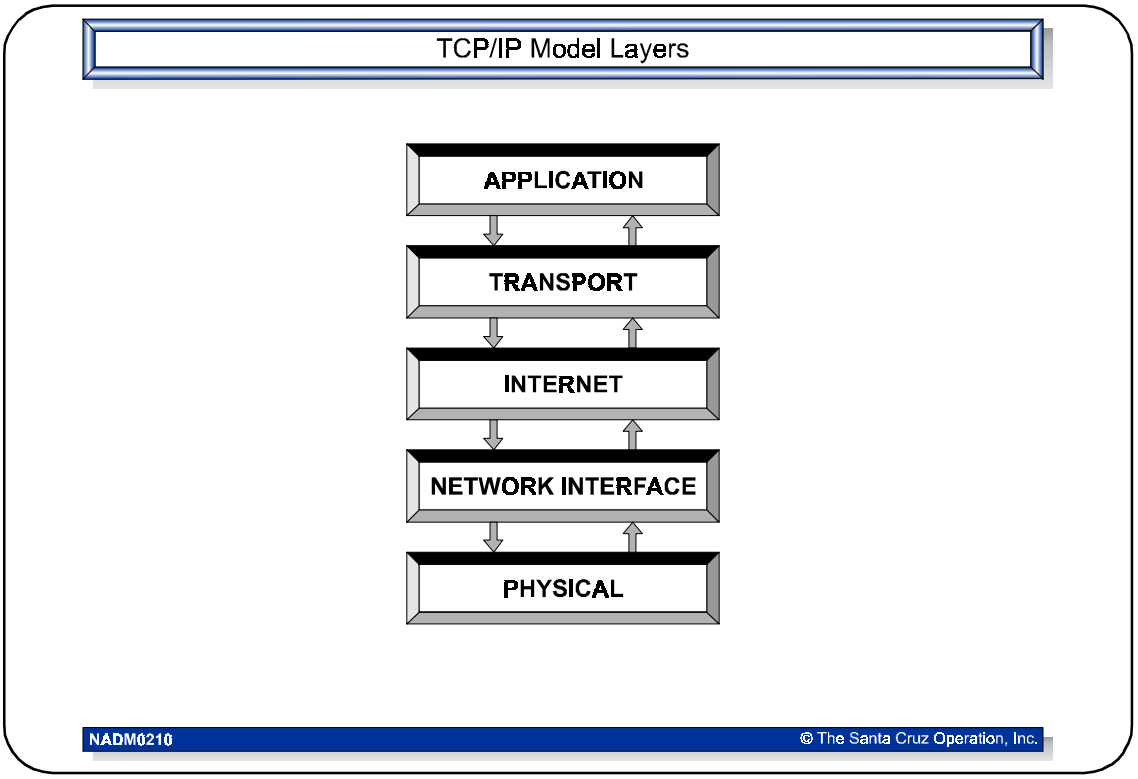
To manage the diversity of tasks required by networks, most networking software divides the requirements into a series of layers.

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


TCP/IP Model Layers
APPLICATION
TRANSPORT
INTERNET
NETWORK INTERFACE
PHYSICAL
NADM0210
© The Santa Cruz Operation, Inc.

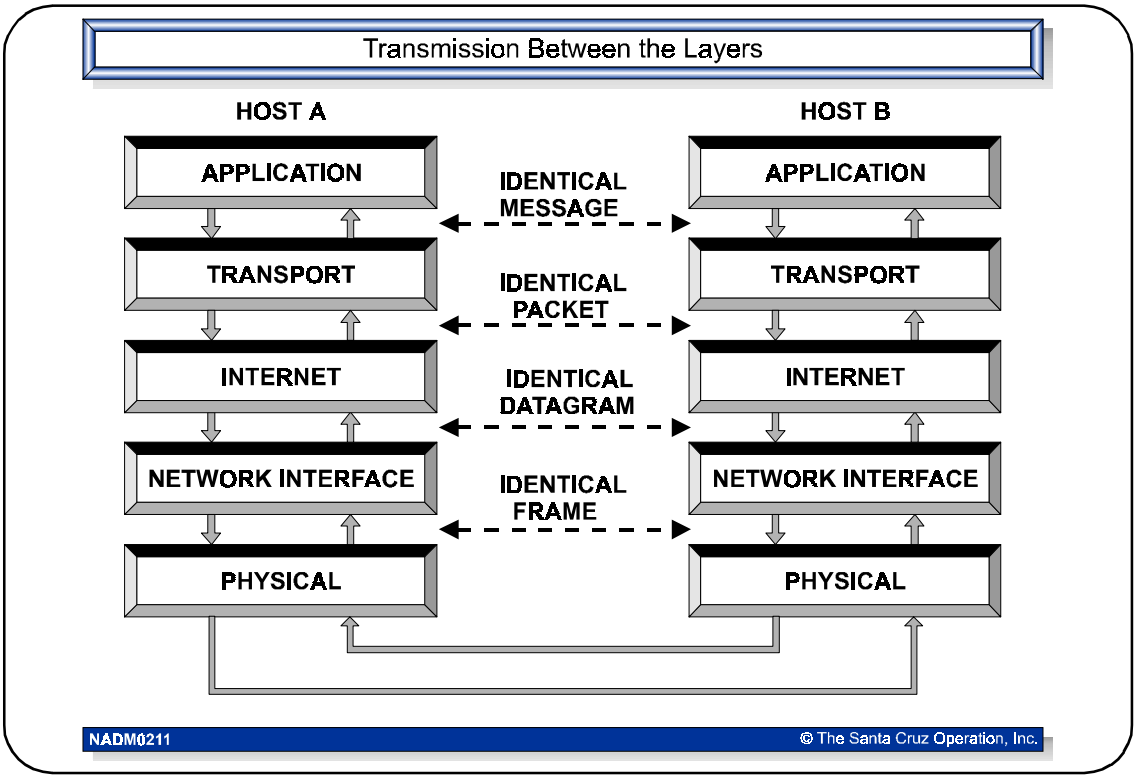
2.7.1 TCP/IP layers

The TCP/IP model uses five layers in its protocol stack. As shown in AV 2-10, these include, in order:

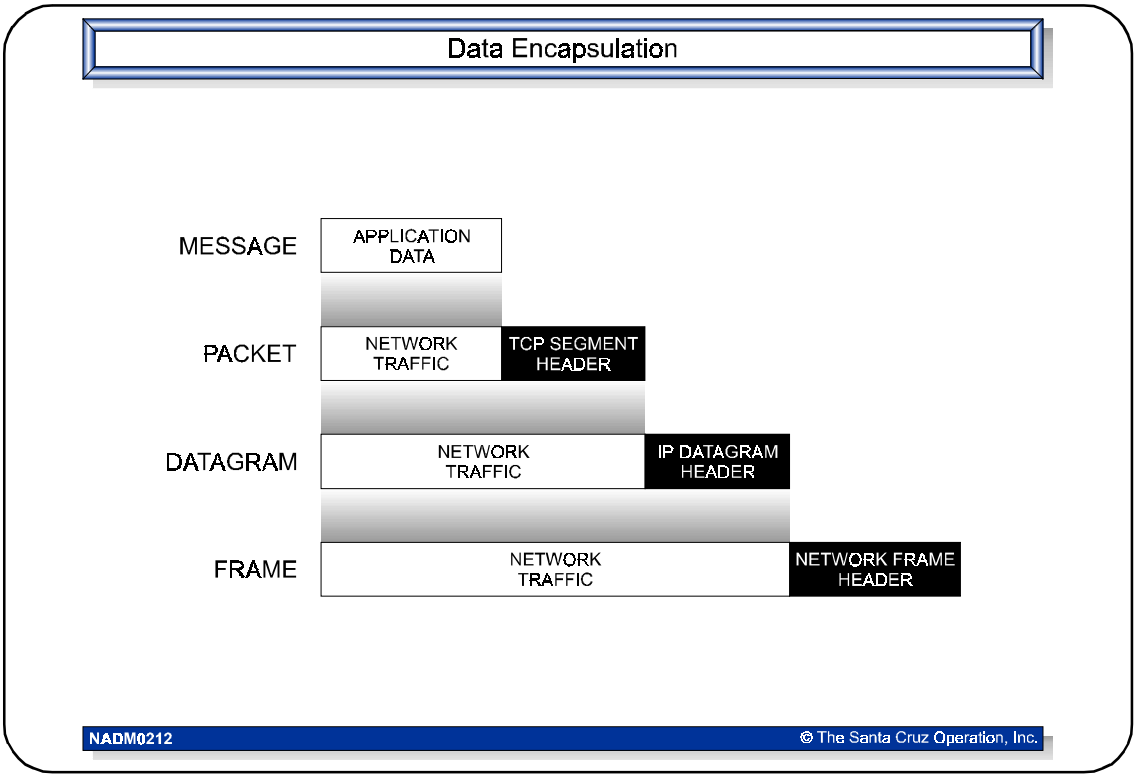
Application layer	The user interacts with the network application. Data comes in as commands from the user and also from network applications on the other end of the connection. TCP/IP applications work in client/server pairs at this layer.
Transport layer	Manages the flow of data between the hosts. The TCP and UDP protocols are implemented at this layer.
Internet layer	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 *network* layer.
Network interface layer	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 *data link* or *link* layer. The drivers supported in SCO OpenServer are discussed in Lesson 3: "Configuring a Network Adapter."
Physical layer	This layer consists of the physical hardware connection between the hosts.

AV 2-10: TCP/IP Model Layers
This model is similar to the Open Systems Interconnection (OSI) 7-layer model which includes the *Presentation* and *Session* layers between the *Application* and *Transport* layers. The functions of the "missing" layers in the TCP/IP model are fulfilled, as needed, by different TCP/IP protocols.

AV 2-11


Transmission Between the Layers
HOST A
HOST B
APPLICATION
TRANSPORT
INTERNET
NETWORK INTERFACE
PHYSICAL
IDENTICAL MESSAGE
IDENTICAL PACKET
IDENTICAL DATAGRAM
IDENTICAL FRAME
APPLICATION
TRANSPORT
INTERNET
NETWORK INTERFACE
PHYSICAL
NADM0211
© The Santa Cruz Operation, Inc.

AV 2-12


Data Encapsulation
MESSAGE
APPLICATION DATA
PACKET
NETWORK TRAFFIC
TCP SEGMENT HEADER
DATAGRAM
NETWORK TRAFFIC
IP DATAGRAM HEADER
FRAME
NETWORK TRAFFIC
NETWORK FRAME HEADER
NADM0212
© The Santa Cruz Operation, Inc.

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 is 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.

The data in a message has meaning only to the application.

- *Packets* are passed between the transport and Internet layers.
- *IP datagrams* are passed between the Internet and network interface layers.
- Network specific *frames* are passed between the network interface and the 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.

AV 2-11: Transmission Between the Layers
This slide shows a message being sent from one system to another as it passes through each layer. Note that Layer *n* on Host B receives exactly the same object that Layer *n* on Host A sent.

AV 2-12: Data Encapsulation
Use this slide to show how the data is encapsulated as it moves from one layer to the next. At each layer, another header is added to the original message (or stripped off on the receiving end).

2.7.3 An example transmission

The **rlogin**(TC) command illustrates how the TCP/IP protocol stack functions. Its syntax is:

```
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**(ADMN) 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 so this:

- The **rlogin** command needs to know which port the **rlogind** (server) is using for network I/O, so it looks up the login service in the */etc/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 Network interface layer

The network interface 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 accepts the frame and passes 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 the media (cables) that connects the systems together in the network.

2.8 LAN Hardware

Section 2.7.3.5, “Physical layer,” on page 2-53 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 provides a brief overview of these LAN technologies:

- Ethernet—A wide range of network adapters are supported by SCO OpenServer.
- IBM Token Ring—A selection of network adapters are supported by SCO OpenServer.
- Fiber Distributed Data Interface (FDDI)—A few network adapters supported by SCO OpenServer.
- 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 2-27 and Lesson 5: “Routing.”

Instructor Guide

2.8.1 Ethernet

Ethernet is the most commonly-used physical layer today. 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:

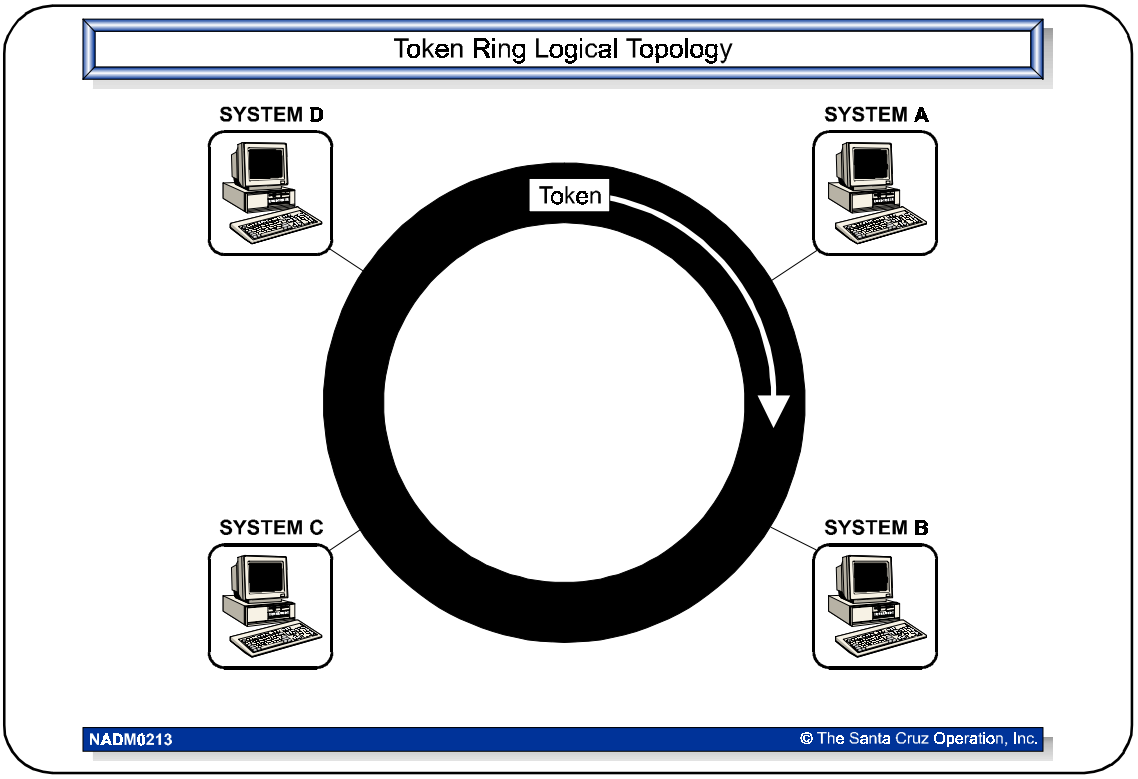
10BASE5 (Thicknet)	Uses 1/2 to 3/8ths inch coaxial cable. Hosts connecting to thicknet must also have external transceivers. Thicknet transmits at 10Mbps and is infrequently used today. Thicknet is limited to 500 meters per segment; maximum 5 segments; 2500 meters maximum cable; maximum 100 stations 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.

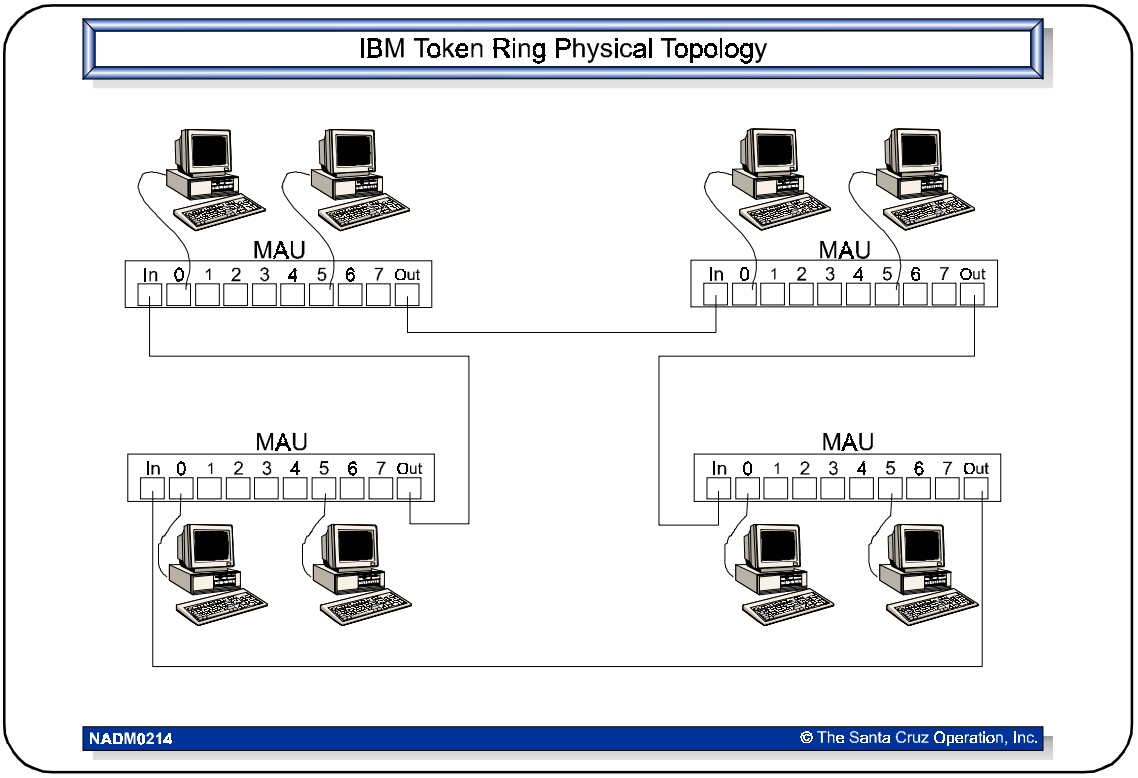
A common optimization in Ethernet LANs is the use of switches because they dramatically decrease collisions. For more information on switches, see Section 2.3, "Network-Joining Devices," on page 2-27.

The IEEE 802.3 "5-4-3" rule specifies there may be no more than 5 segments, 4 repeaters, and 3 populated segments; hence the maximum stations of 300 (10BASE5) and 90 (10BASE2).

AV 2-13


Token Ring Logical Topology
SYSTEM D
SYSTEM A
Token
SYSTEM C
SYSTEM B
NADM0213
© The Santa Cruz Operation, Inc.

AV 2-14


IBM Token Ring Physical Topology
MAU
In 0 1 2 3 4 5 6 7 Out
MAU
In 0 1 2 3 4 5 6 7 Out
MAU
In 0 1 2 3 4 5 6 7 Out
MAU
In 0 1 2 3 4 5 6 7 Out
NADM0214
© The Santa Cruz Operation, Inc.

2.8.2 IBM Token Ring

AV 2-13: Token Ring Logical Topology
Use this slide to illustrate the logical topology of a Token Ring network.

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

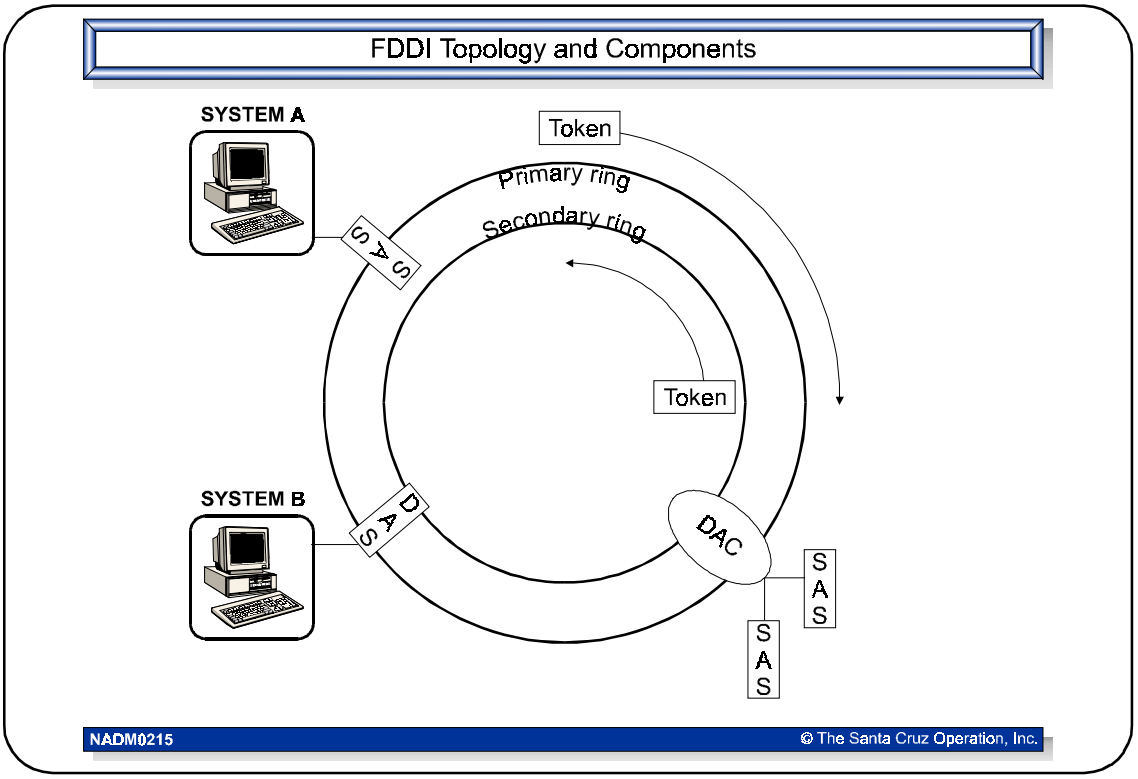
Each workstation is connected to a Multiple Access Unit (MAU), which supports up to eight workstations. Additional MAUs can be cascaded to support more workstations, as illustrated in AV 2-14.

Token Ring runs at speeds of either 4 or 16 Mbps and is subject to these cabling limitations:

- 45 meters between an MAU and a workstation
- 45 meters between two MAUs
- 120 meters patch cable connecting all MAUs
- 12 MAUs for a total of 96 total stations.

AV 2-14: IBM Token Ring Physical Topology
Use this slide to illustrate how IBM token rings are physically put together.

AV 2-15


FDDI Topology and Components
SYSTEM A
Token
Primary ring
Secondary ring
S A S
Token
SYSTEM B
S A S
DAC
S A S
S A S
NADM0215
© The Santa Cruz Operation, Inc.

2.8.3 FDDI

AV 2-15: FDDI Topology and Components
Use this slide to illustrate the topology and components of 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)

NOTE: IPv6 is sometimes called IPNG (IP, the next generation). Technical specifications for IPv6 can be found at *http://playground.sun.com*.

The Internet Protocol described 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 that 10^9 addresses. But because of the way they are allocated by the InterNIC, there are not enough to meet the demand.
- 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

Instructor Guide

This is an immense number. For example, it is large enough to assign every person on the earth 10^{28} addresses. Make sure students realize that this is far greater than 4 times the current address space.

- 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 to the ISPEC encryption specification

In other words, the concept of secure, encrypted data is built into the protocol definition. It does not have to be back-patched using methods like SSL as it does in IPv4.

Check the SCO Web page for information about plans to support IPv6 in SCO products when it becomes a ratified standard.

As of the writing of this course, IPv6 has not been fully ratified as a standard.

2.10 Lesson Summary

The SCO OpenServer 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 does the netmask do?

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 the address resolution protocol (ARP)? Describes 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/services* files?

2.11 Relevant Commands, Files, and Syntax

The following commands and files were presented in this lesson.

Relevant commands

arp(ADMN)	Displays and modifies the Internet-to-Ethernet address translation table used for the address resolution protocol (ARP)
ndstat(ADM)	Displays statistics for configured network adapter drivers
rlogin(TC)	Is the remote login client, that connects your terminal on the current local host to a remote host system
rlogind(ADMN)	Is the server for the **rlogin**(TC) program
inetd(ADMN)	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

/etc/inetd.conf	Is the configuration file for **inetd**(ADMN). See also *inetd.conf*(SFF).
/etc/hosts	Lists the host names and IP addresses of the hosts that share the network. See *hosts*(SFF).
/etc/services	Lists the service name, port number, protocol name, and aliases of well-known network services. See *services*(SFF).

2.12 Answers to Excercises and Review Questions

This section provides suggested answers to exercises and review questions.

Exercise: Counting nodes on the network

The maximum number of nodes that *could* exist on a network given a 4-byte (32-bit) address is 2^{32} = 4,294,967,296. The *actual* number is smaller because some values are reserved.

Exercise: Specifying netmasks

- Netmask for Class A network: 255.0.0.0
- Netmask for Class C network: 255.255.255.0

Exercise: Determining broadcast addresses

- 85.110.43.7 is a Class A address that uses the broadcast address 85.255.255.255
- 210.20.210.20 is a Class C address that uses the broadcast address 210.20.210.255
- 150.75.37.5 is a Class B address that uses the broadcast address 150.75.255.255

Exercise: Identifying ports and protocols

- **netstat**—Port 15, TCP protocol
- **ftp**—Port 21, TCP protocol
- **who**—Port 513, UDP protocol
- **login**—Port 513, TCP protocol

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 include:

   - 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
- Network interface layer (also called data link or link 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/services* file lists the service name, port number, protocol name, and aliases of well-known network services.

3

Configuring a Network Adapter

Instructor Guide

Lesson 3

Configuring a Network Adapter

Suggested lesson duration

This lesson should take approximately 2 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

Ideally, the student systems should be set up from a fresh installation without the networking interface configured. If the systems have been used in previous classes, verify that networking is not currently configured:

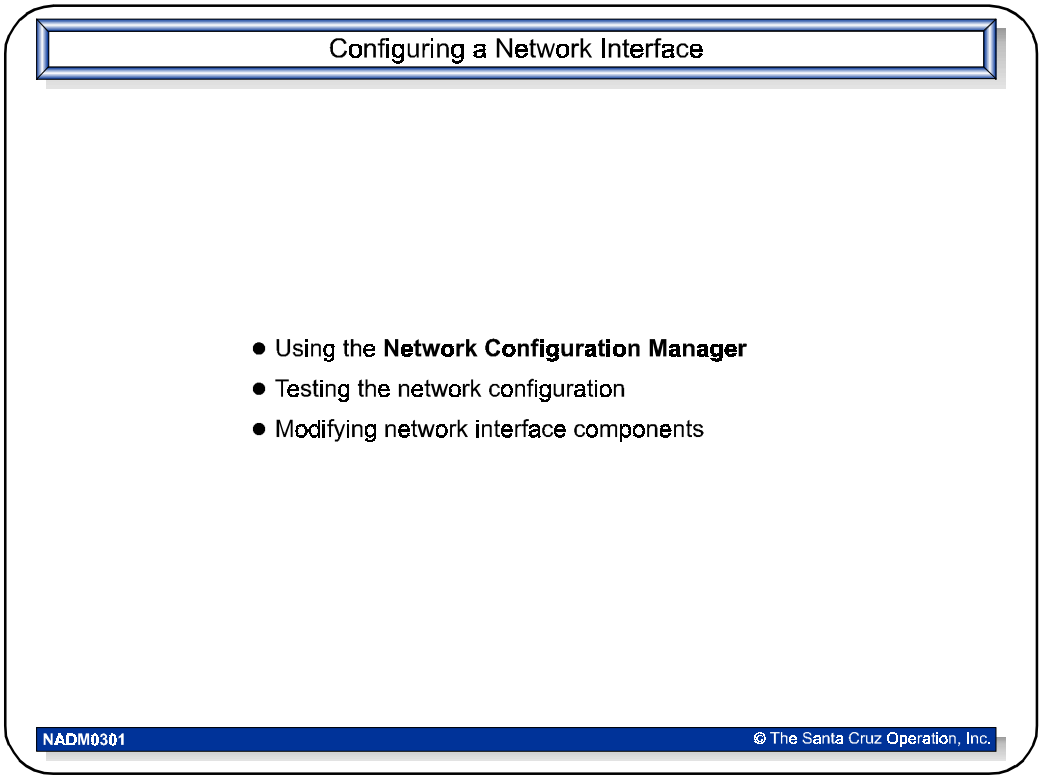
- If necessary, use **netconfig**(ADM) to remove the network interface. You might want to have the students do this.
- Remove all entries from */etc/hosts* except the `localhost` entry and the entry with the local machine name. If the there is no network card configured, set the local host name to resolve to the loopback address.
- Remove the DNS client configuration, if present.
- Assemble all the necessary hardware information for the machines being used.

Instructor preparation

This lesson describes the hardware attributes of network adapters. You might want to review this material with the appropriate hardware references, including:

- Network adapter documentation
- The **Intel® *n86 Microprocessor Programmer's Reference Manual***. The basic issues are the same for the i386™, i486™, and Pentium® CPUs, so any of these reference manuals will suffice.
- Kettle, Peter and Steve Statler, ***Writing Device Drivers for SCO UNIX: A Practical Approach***, Wokingham, England: Addison-Wesley, 1992, ISBN: 0-201-54425-3. This book covers basic hardware issues as well as how to write a device driver.

AV 3-1


Configuring a Network Interface
Using the **Network Configuration Manager**
Testing the network configuration
Modifying network interface components
NADM0301
© The Santa Cruz Operation, Inc.

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.

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**(ADM) **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
- Add, modify, or remove network interface components

AV 3-1: Configuring a Network Interface
The first slide describes the overall topics covered in this lesson.

3.1 Configuring Network Interfaces

Networking is enabled on an SCO OpenServer 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 3-9.

3.1.1 Network hardware

SCO OpenServer supports a number of common PCI, ISA, EISA, and Plug-and-Play network adapters. 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:

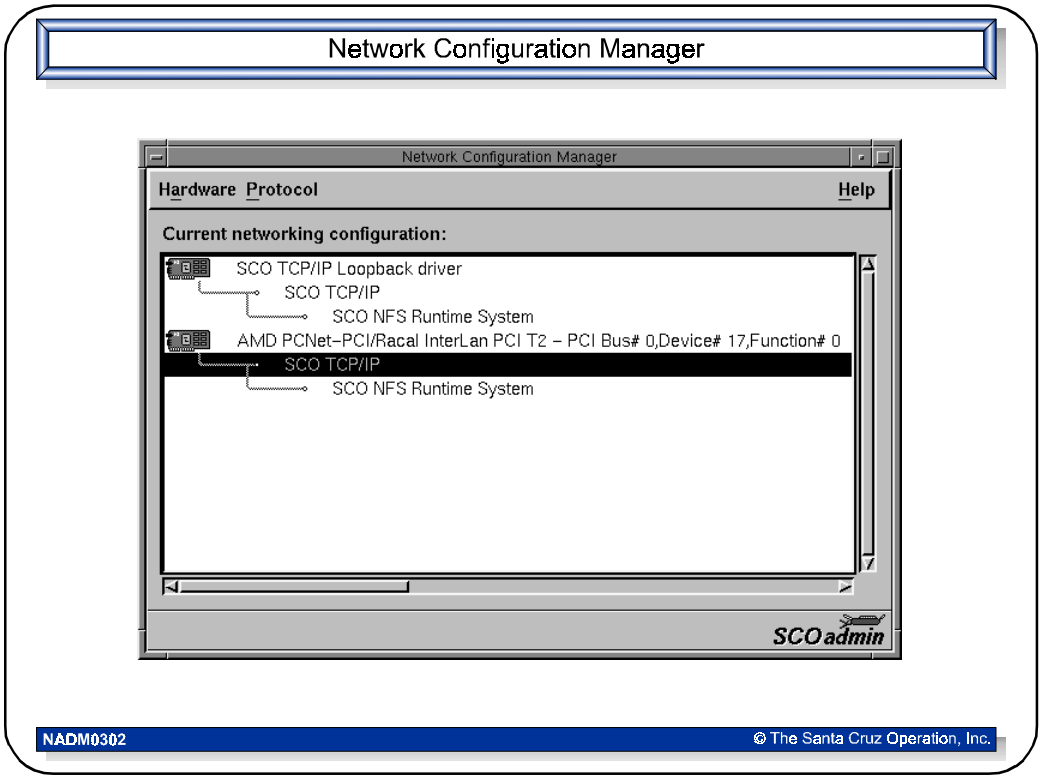
- IRQ
- I/O base address
- RAM base address
- RAM size of the network card
- DMA channel

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

Supported network adapters are listed in **scohelp ⇨ SCO OpenServer Features and Limitations ⇨ Supported network adapters**. The latest list of supported hardware can be found on the Compatible Hardware Web Pages at:

http://wdb1.sco.com/chwp/owa/hch_search_form

AV 3-2


Network Configuration Manager
Network Configuration Manager
Hardware Protocol
Help
Current networking configuration:
SCO TCP/IP Loopback driver
SCO TCP/IP
SCO NFS Runtime System
AMD PCNet–PCI/Racal InterLan PCI T2 – PCI Bus# 0,Device# 17,Function# 0
SCO TCP/IP
SCO NFS Runtime System
SCOadmin
NADM0302
© The Santa Cruz Operation, Inc.

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. SCO OpenServer includes drivers for PCI, ISA, EISA, and Plug-and-Play network adapter cards. These drivers make up the network interface layer discussed in Section 2.7.1, “TCP/IP layers,” on page 2-49.

Two types of network adapter drivers are supported:

MDI — MAC Driver Interface is the current driver technology used in SCO OpenServer.

LLI — Link Level Interface driver technology is used in older versions of SCO OpenServer.

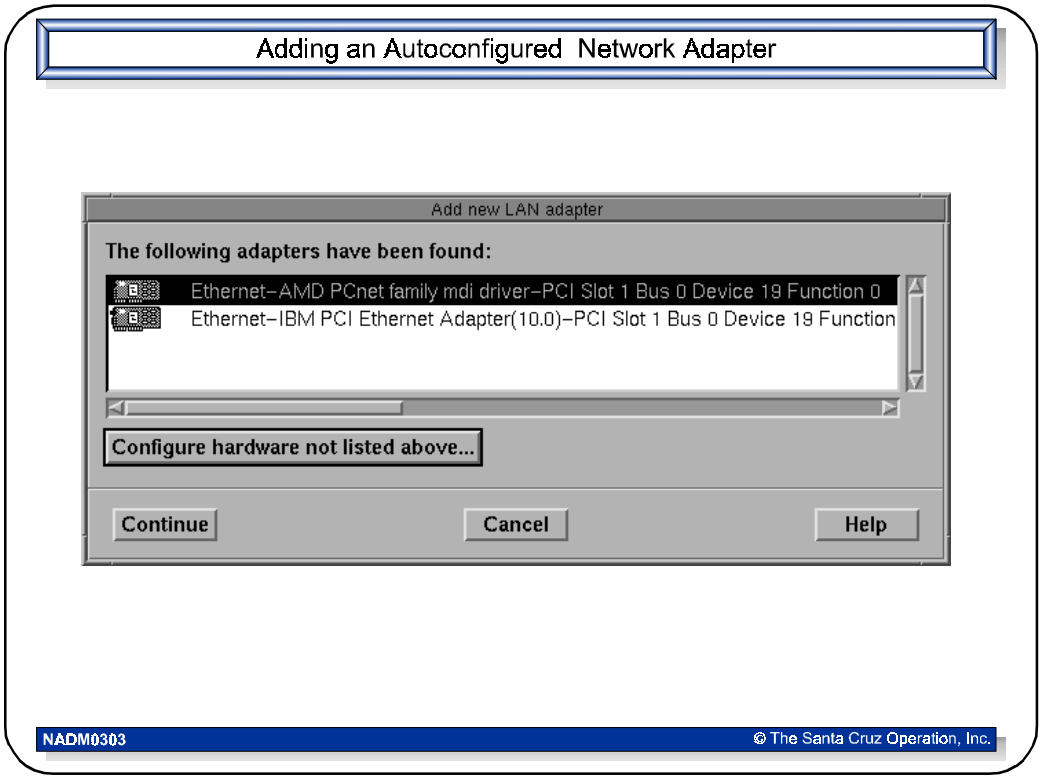
Not all networking features are supported on each driver type. For example, automatic detection is only supported on network adapters with MDI drivers.

AV 3-2: Network Configuration Manager
This slide shows an example network interface configured with the **Network Configuration Manager**. Explain that each component represents one of the links of the chain.

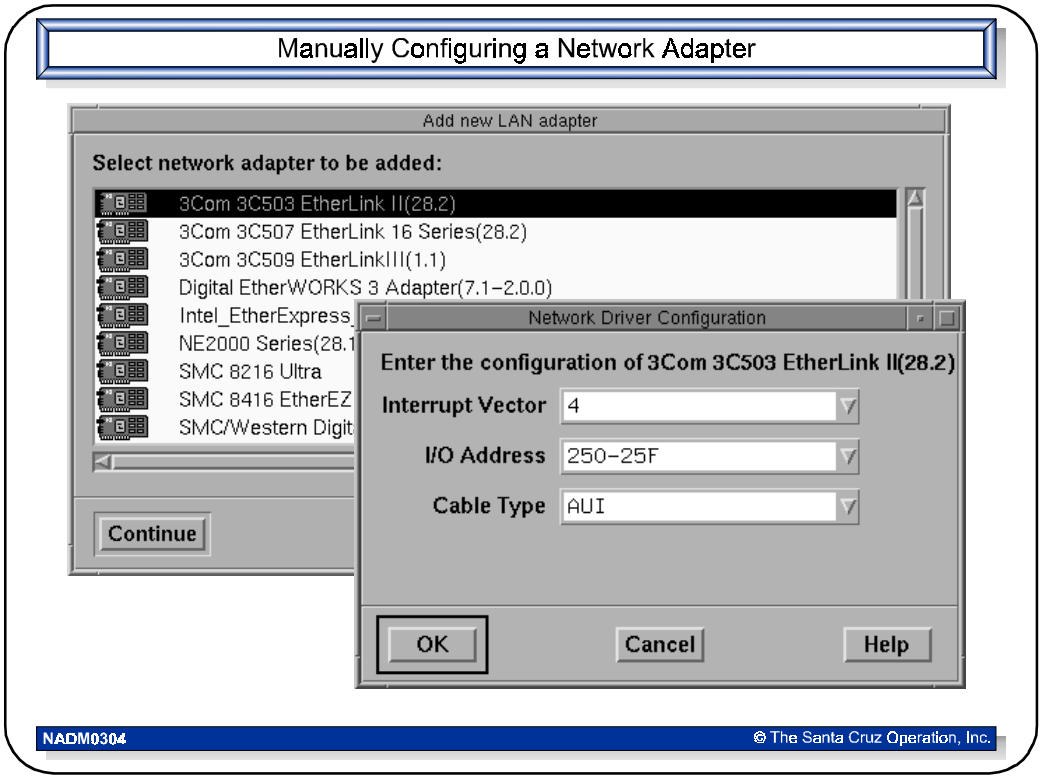
The **Network Configuration Manager** is a tool that allows you to configure network adapter drivers and manage network connections. Remember, the **Network Configuration Manager** can be started in any of the following ways:

- From **SCOadmin**, select **Networks** ⇨ **Network Configuration Manager**.
- From the command line, enter **`netconfig`**.
- From the command line, enter **`scoadmin network config`**.

AV 3-3


Adding an Autoconfigured Network Adapter
Add new LAN adapter
The following adapters have been found:
Ethernet–AMD PCnet family mdi driver–PCI Slot 1 Bus 0 Device 19 Function 0
Ethernet–IBM PCI Ethernet Adapter(10.0)–PCI Slot 1 Bus 0 Device 19 Function
Configure hardware not listed above...
Continue
Cancel
Help
NADM0303
© The Santa Cruz Operation, Inc.

AV 3-4


Manually Configuring a Network Adapter
Add new LAN adapter
Select network adapter to be added:
3Com 3C503 EtherLink II(28.2)
3Com 3C507 EtherLink 16 Series(28.2)
3Com 3C509 EtherLinkIII(1.1)
Digital EtherWORKS 3 Adapter(7.1–2.0.0)
Intel_EtherExpress
NE2000 Series(28.1
SMC 8216 Ultra
SMC 8416 EtherEZ
SMC/Western Digit
Continue
Network Driver Configuration
Enter the configuration of 3Com 3C503 EtherLink II(28.2)
Interrupt Vector
4
I/O Address
250–25F
Cable Type
AUI
OK
Cancel
Help
NADM0304
© The Santa Cruz Operation, Inc.

3.1.2.1 Adding a network adapter

AV 3-3: Adding an Autoconfigured Network Adapter
This screen is displayed when a LAN card is autodetected.

To add a network adapter:

Step 1. Start the **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, Select the **TCP/IP** protocol.
- 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.

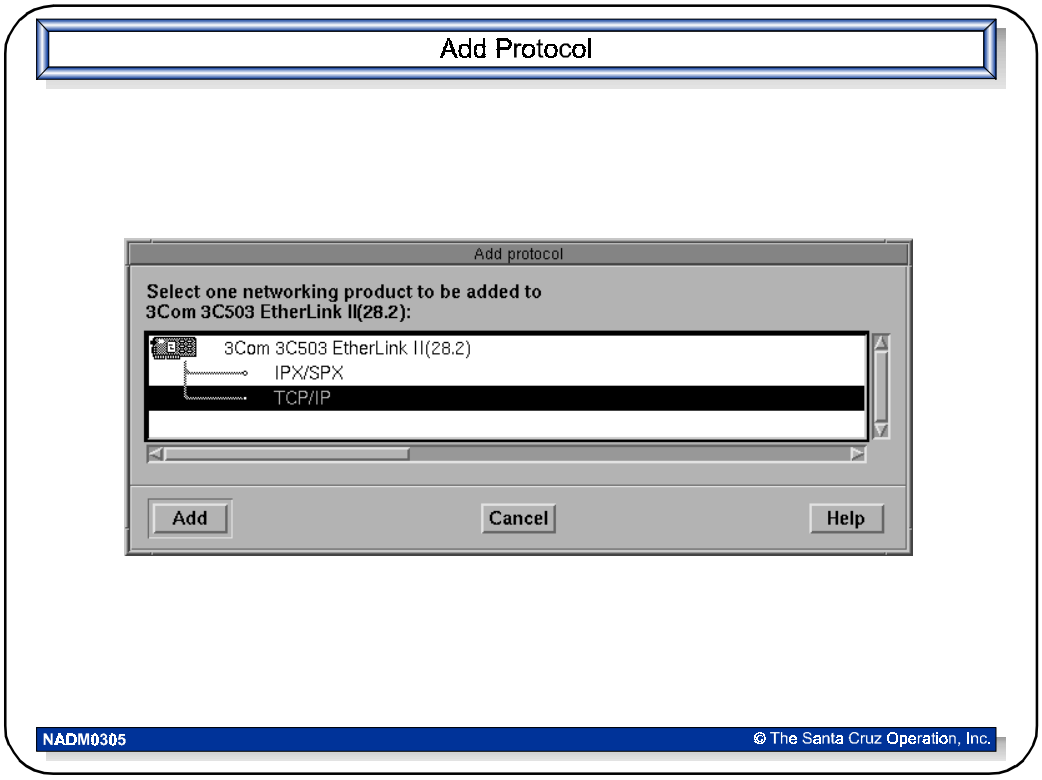
AV 3-4: Manually Configuring a Network Adapter
This slide shows the series of screens used to configure a non-autodetected network adapter.

Step 4. Select an adapter from the list of supported adapters (arranged by vendor), or, if prompted, choose to search for your adapter. Click on **Continue**.

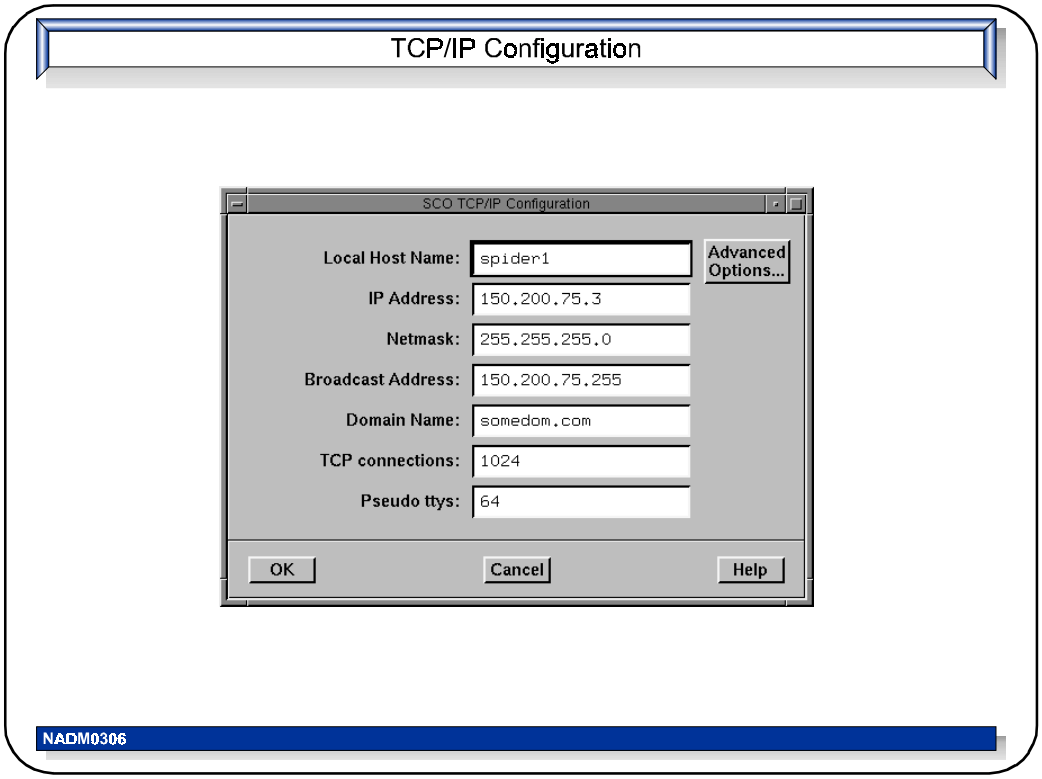
Step 5. If prompted, click on **Yes** to search for the selected adapter then click on **Continue** if the adapter is found.

Step 6. 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


Add Protocol
Add protocol
Select one networking product to be added to
3Com 3C503 EtherLink II(28.2):
3Com 3C503 EtherLink II(28.2)
IPX/SPX
TCP/IP
Add
Cancel
Help
NADM0305
© The Santa Cruz Operation, Inc.

AV 3-6


TCP/IP Configuration
SCO TCP/IP Configuration
Local Host Name:
spider1
Advanced Options...
IP Address:
150.200.75.3
Netmask:
255.255.255.0
Broadcast Address:
150.200.75.255
Domain Name:
somedom.com
TCP connections:
1024
Pseudo ttys:
64
OK
Cancel
Help
NADM0306

Step 7. 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.

AV 3-5: Add Protocol
This slide shows a screen shot of the **Add Protocol** window.

Step 8. Select the **TCP/IP** protocol and click on **Add**.

Step 9. Enter the appropriate values for the displayed fields when the **SCO TCP/IP Configuration** window is displayed.

Local Host Name	Enter the name by which this system will be known on the network.
IP address	Assign an IP address to this network node.
Netmask	Enter the appropriate netmask.
Broadcast address	Enter the broadcast address.
Domain Name	Enter the name of the domain to which this machine belongs.
TCP connections	Enter the maximum number of allowed TCP/IP connections.
Pseudo ttys	Enter the number of pseudo-ttys configured. Pseudo-ttys enable users on other hosts to use **telnet** or **rlogin** to access your host. The default, 64, is adequate for several (2-10) users to access your system simultaneously.

Step 10. Click on **OK** when finished.

AV 3-6: TCP/IP Configuration
This slide shows a screen shot of the **SCO TCP/IP Configuration** window accessed from the **Network Configuration Manager**.

Ignore the **Advanced options** button for now. The use of this button is discussed in Section 5.1.1, “Installing additional network adapters,” on page 5-9.

Step 11. Click on **OK** to acknowledge the configuration of the protocol.

Step 12. Exit from the **Network Configuration Manager**. Select **Hardware ⇨ Exit**.

Step 13. Click on **Yes** to relink the UNIX kernel.

Step 14. Answer '**y**' to boot the new kernel by default and rebuild the kernel environment. When finished, press ENTER.

Step 15. Reboot the system.

For additional information on configuring network adapters, refer to **scohelp ⇨ Networking Documentation Set ⇨ Configuring Network Connections ⇨ Chapter 1: Configuring network connections.**

3.2 Testing TCP/IP Connectivity

Note that this is not a troubleshooting section. The goal is to show students simple ways to verify that the networking interface has been configured correctly, not to teach them how to correct problems.

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

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

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.

This is the first mention of the term *loopback interface*. It is introduced here because students will see it when running **netstat**. It is covered more fully in Section 3.2.3 on page 3-17.

3.2.1 The netstat(TC) command

This section is not intended as an exhaustive discussion of the **netstat** command, but rather a brief introduction to a useful monitoring tool. Encourage students to find out more about **netstat** on their own.

The **netstat**(TC) 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

"Auto-configured" means those interfaces that are detected and configured when the system boots. This does not include those interfaces that may happen to be present, but not 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.

Both options may be specified as one command-line argument (**-in**). Emphasize the usefulness of **-n** when the network is unreachable.

The **netstat** output displays the `loopback` interface, *lo0*, plus one interface for each networking card or serial line interface configured on your system. The *atl0* interface is also displayed. This is a placeholder interface used for certain virtual domain configurations.

If no interfaces are displayed, TCP/IP has not been configured correctly. Refer to troubleshooting information in s**cohelp ⇨ Networking Documentation Set ⇨ Configuring Network Connections ⇨ Chapter 1: Configuring network connections ⇨ Troubleshooting network configuration.**

The `loopback` interface is discussed in Section 3.2.3 on page 3-17.

3.2.2 The ping(ADMN) command

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

The basic syntax is:

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

By default, the **ping** command sends one datagram per second, prints one line of output for every response that it receives, and calculates packet loss statistics.

Use **ping** to contact hosts on the local network running TCP/IP:

```
ping cousin
PING 150.200.76.40 (150.200.76.40): 56 data bytes
64 bytes from nec.curric.com (150.200.76.40): icmp_seq=0 ttl=32 time=0 ms
64 bytes from nec.curric.com (150.200.76.40): icmp_seq=1 ttl=32 time=0 ms
64 bytes from nec.curric.com (150.200.76.40): icmp_seq=2 ttl=32 time=0 ms
64 bytes from nec.curric.com (150.200.76.40): icmp_seq=3 ttl=32 time=0 ms
64 bytes from nec.curric.com (150.200.76.40): icmp_seq=4 ttl=32 time=0 ms
```

When you press the interrupt key (often DEL or CTRL c) to stop sending packets, **ping** displays packet loss statistics:

```
--- 150.200.76.40 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0/0/0 ms e
```

When used with a broadcast address, **ping** is a convenient way to verify connectivity to all hosts on your network:

```
ping 150.200.76.255
```

Be careful using **ping** with broadcast addresses if you are connected to the Internet.

Remind students that "all the hosts on your network" refers only to those local hosts on the internetwork (the subnet)—not every host on every network.

Use the LCD to show students how to use **ping**. Emphasize how to use the *interrupt* key to break out of the loop and get summary statistics.

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**(XC), 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 do 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**(ADMN) 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(ADMN) and netstat(TC)

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

Test the TCP/IP software. Enter:

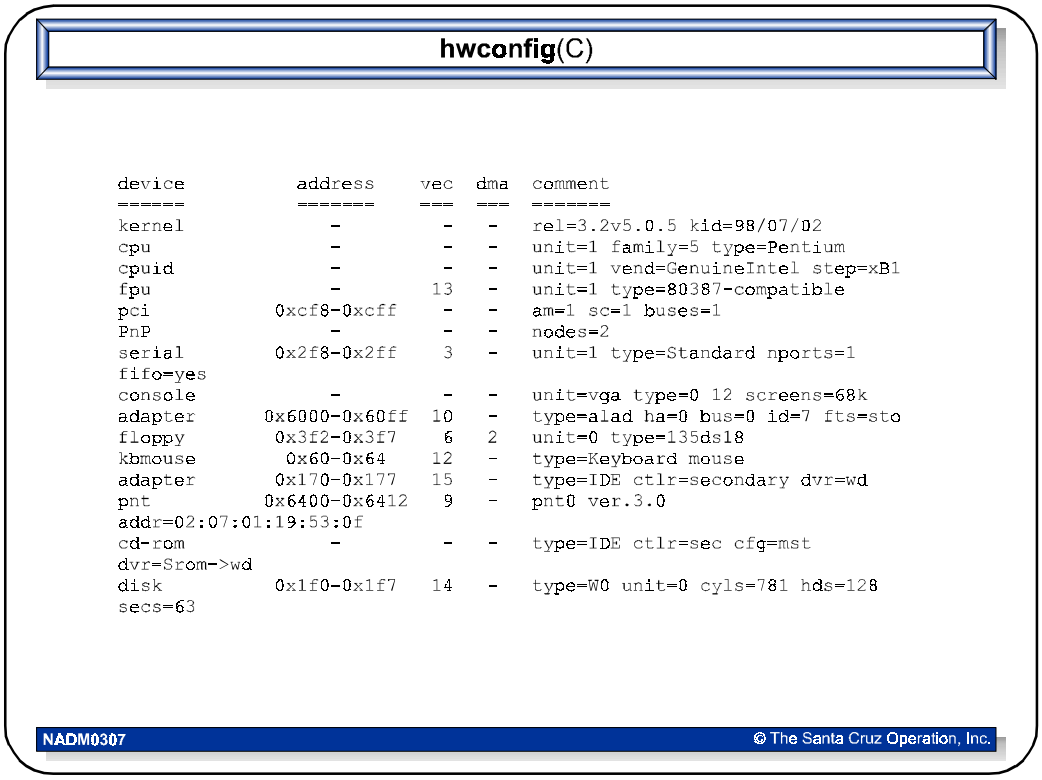
```
ping localhost
```

After you stop **ping**, use **netstat** with the **-i** option to determine the number of packets transmitted on the *lo0* interface. Did the number of transmitted packets increase?

Instructor Guide

This might be a good time to do the first part of the lab in Section 3.5.1, “Configuring a network adapter,” on page 3-26. The sections that follow assume students have a working network adapter.

AV 3-7


hwconfig(C)
device address vec dma comment
kernel - - - rel=3.2v5.0.5 kid=98/07/02
cpu - - - unit=1 family=5 type=Pentium
cpuid - - - unit=1 vend=GenuineIntel step=xB1
fpu - 13 - unit=1 type=80387-compatible
pci 0xcf8-0xcff - - am=1 sc=1 buses=1
PnP - - - nodes=2
serial 0x2f8-0x2ff 3 - unit=1 type=Standard nports=1
fifo=yes
console - - - unit=vga type=0 12 screens=68k
adapter 0x6000-0x60ff 10 - type=alad ha=0 bus=0 id=7 fts=sto
floppy 0x3f2-0x3f7 6 2 unit=0 type=135ds18
kbmouse 0x60-0x64 12 - type=Keyboard mouse
adapter 0x170-0x177 15 - type=IDE ctlr=secondary dvr=wd
pnt 0x6400-0x6412 9 - pnt0 ver.3.0
addr=02:07:01:19:53:0f
cd-rom - - - type=IDE ctlr=sec cfg=mst
dvr=Srom->wd
disk 0x1f0-0x1f7 14 - type=W0 unit=0 cyls=781 hds=128
secs=63
NADM0307
© The Santa Cruz Operation, Inc.

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 3-17), 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 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**(TC) 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 `Oerrs` indicates that the I/O address for the network card might be incorrect.
- `Ipkts` is zero and `Opkts` equals `Oerrs` indicates that the network cable might have a faulty connection.

Use the LCD to demonstrate `netstat -i`. Watch the statistics over time for the instructor machine's interface.

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 */usr/adm/messages* and can be viewed using the **System Logs Manager**.
- Use the **hwconfig**(C) command to view the present hardware configuration.

AV 3-7: hwconfig(C)
This slide shows an example of the hardware configuration information displayed by **hwconfig**(C).

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.

A device called a cable scanner—also known as a Time Domain Reflectometer (TDR) can easily pinpoint shorts and openings in the cable, as well as act as an inductive tracer for hard-to-access cables in ceilings and walls. It uses radar technology to echo signals down the cable and records elapsed time for the return signal to determine where cable faults occur. It is fairly expensive, but if you intend to run a complex network or troubleshoot other users' networks, it could be worth the price.

The **ndstat**(ADM) (formerly **llistat**) command can 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, such as:

- Device name
- Hardware address
- Information it receives from polling the DLPI (Data Link Provider Interface) module for information on the services it provides

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

Exercise: Using hwconfig(C) and ndstat(ADM)

Use **hwconfig**(C) to view the present network adapter configuration.

Use **ndstat**(ADM) to view the statistics for your configured network adapter driver.

3.3 Modifying Network Interface Components

After configuring your network adapter, it might be necessary to modify your network interface components 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 might need to add other protocols. For example, you might want to configure a system as a gateway to a Novell-based network using the IPX/SPX protocol.

To add a protocol used with a network adapter:

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

Step 2. Select the network adapter you want to configure.

Step 3. Select **Protocol** ⇨ **Add Protocol**.

Step 4. Select the new protocol from the Add Protocol window.

Step 5. Click on **Add**.

Step 6. Specify the protocol parameters.

Each additional protocol requires different configuration information.

Step 7. Click on **OK** to complete the configuration.

Refer to **scohelp** ⇨ **Networking Documentation Set** ⇨ **Configuring Network Connections** ⇨ **Chapter 1: Configuring network 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.

3.4 Lesson Summary

This lesson described the steps used to configure a network adapter. It also described the procedures to set up the protocols and client services necessary when connecting to a LAN or to the Internet. Some of the 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**(ADMN) and **netstat**(TC) commands
- The `loopback` interface

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. What is the purpose of the **netstat**(TC) command?

3.5 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.

3.5.1 Configuring a network adapter

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 may 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 will also 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 may 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 the **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 the **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.

Step 4. 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 5. If prompted, click on **Yes** to search for the selected adapter then click on **Continue** if the adapter is found.

Step 6. Enter configuration information for your adapter (such as interrupt vector, I/O address) and click on **OK**. The fields might be 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** and click on **Add**.

Step 9. Enter the appropriate values for the displayed fields when the **SCO TCP/IP Configuration** window is displayed.

Local Host Name	Enter the name by which this system will be known on the network.
IP address	Assign an IP address to this network node.
Netmask	Enter the appropriate netmask.
Broadcast address	Enter the broadcast address.
Domain Name	Enter the name of the domain to which this machine belongs.
TCP connections	Enter the maximum number of allowed TCP/IP connections.
Pseudo ttys:	Enter the number of pseudo-ttys configured. Pseudo-ttys enable users on other hosts to use **telnet**(TC) or **rlogin**(TC) to access your host. The default, 64, is adequate for several (2-10) users to access your system simultaneously.

Step 10. Click on **OK** when finished.

Step 11. Click on **OK** to acknowledge the configuration of the protocol.

Step 12. Select **Hardware ⇨ Exit** to exit the **Network Configuration Manager**.

Step 13. Click on **Yes** to relink the UNIX kernel.

Step 14. Answer '**y**' to boot the new kernel by default and rebuild the kernel environment. When finished, press ENTER.

Step 15. Reboot the system.

The network adapter is now configured and ready to use.

Test the network adapter:

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

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

3.6 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

hwconfig(C)	Displays the current hardware configuration
ndstat(ADM)	Displays network adapter driver statistics
netconfig(ADM)	Configures network interfaces
netstat(TC)	Displays the contents of various network-related data structures
ping(ADMN)	Tests network connections by eliciting hosts to reply to packets sent to them

3.7 Answers to Excercises and Review Questions

This section provides suggested answers to exercises and review questions.

Exercise: Using ping(ADMN) and netstat(ADMN)

The number of transmitted packets should have increased.

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**(ADMN) command to test the software configuration. Enter: **`ping localhost`**.

4. The **netstat**(TC) command allows you to examine the currently configured interfaces and other network-related data structures.

4

Subnets

Instructor Guide

Lesson 4

Subnets

Suggested lesson duration

This lesson should take approximately 2 hours to complete. There is no lab exercise.

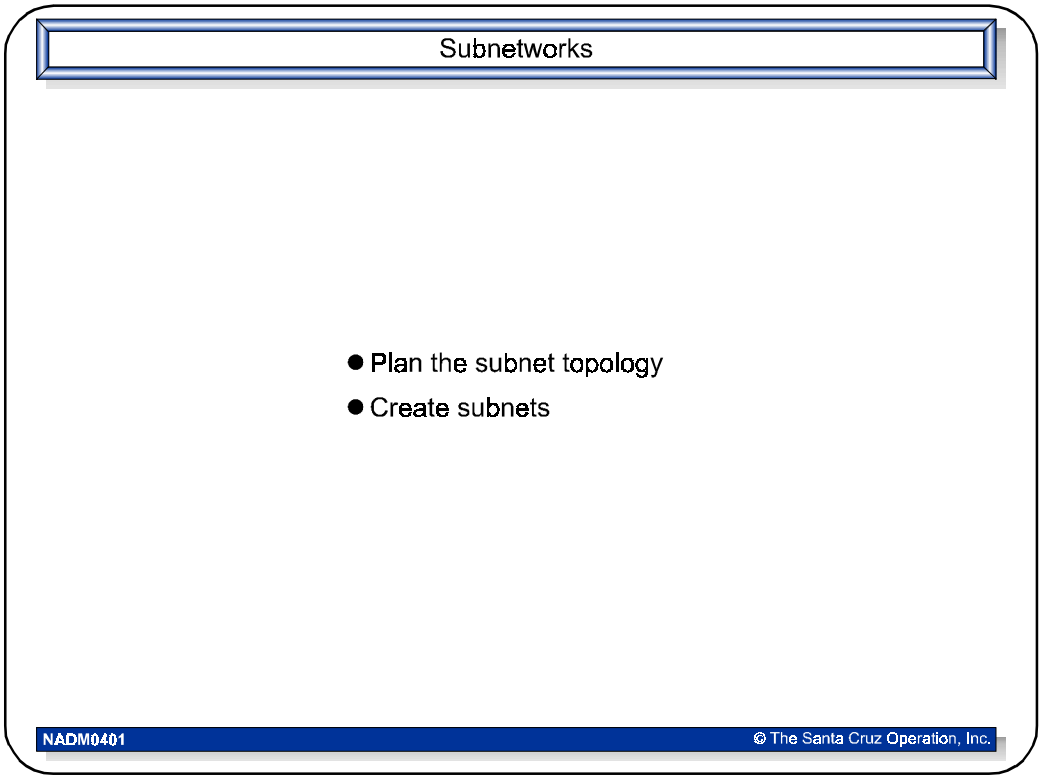
Pre-lesson classroom preparation

None.

Instructor preparation

You must be thoroughly familiar with the functions of various components of an interface's address configuration including address classes, netmask function, and broadcast addresses. You must also be familiar with the reasons for creating subnets, such as address allocation, performance, and security.

AV 4-1


Subnetworks
Plan the subnet topology
Create subnets
NADM0401
© The Santa Cruz Operation, Inc.

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.

Networks are often divided into subnets to enable improvements in network management, such as:

- Simpler administration
- Enhanced security
- Improved performance
- Expansion
- Troubleshooting

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

AV 4-1: Subnets
The first slide describes the overall topics covered in this lesson.

4.1 Planning a Subnet Topology

Have the students describe their current network topology to you. Ask them what their concerns are for their network's near and long-term future. Use this knowledge as you present and highlight the issues discussed in this section.

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:

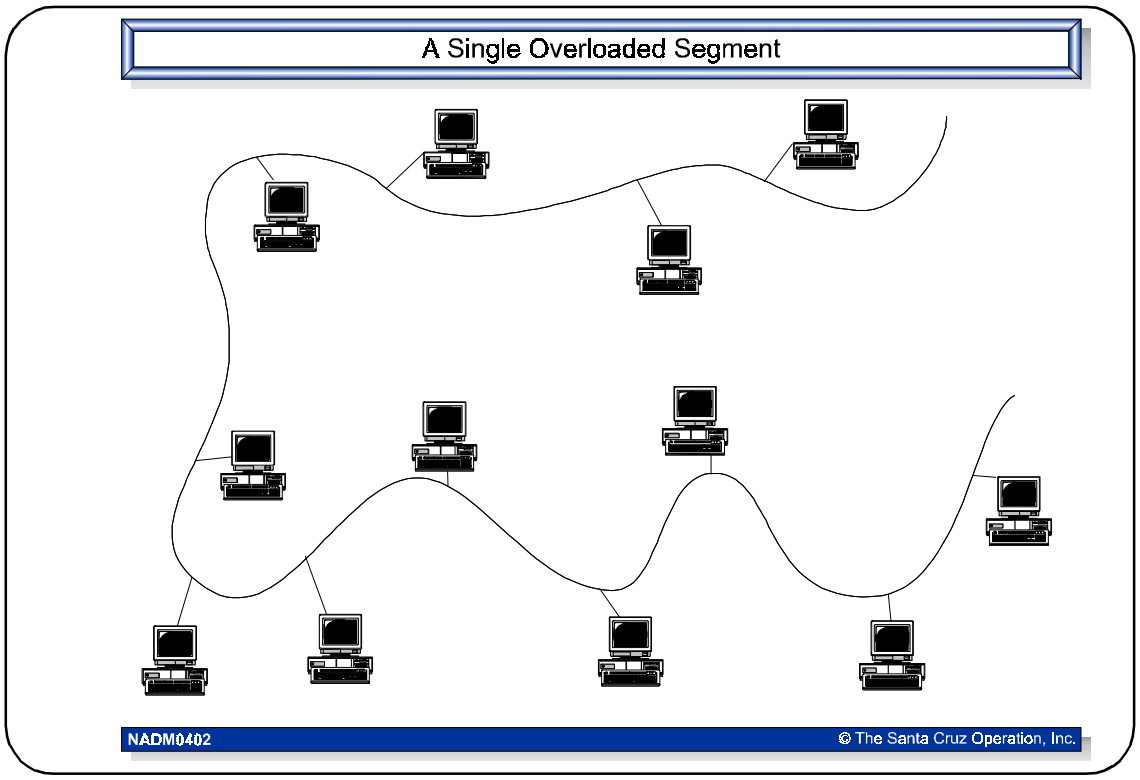
- An increase in the number of nodes
- A need to combine several independent LANs into a single enterprise LAN
- Increased traffic as services become more widely dispersed

Other advantages of creating subnetworks include:

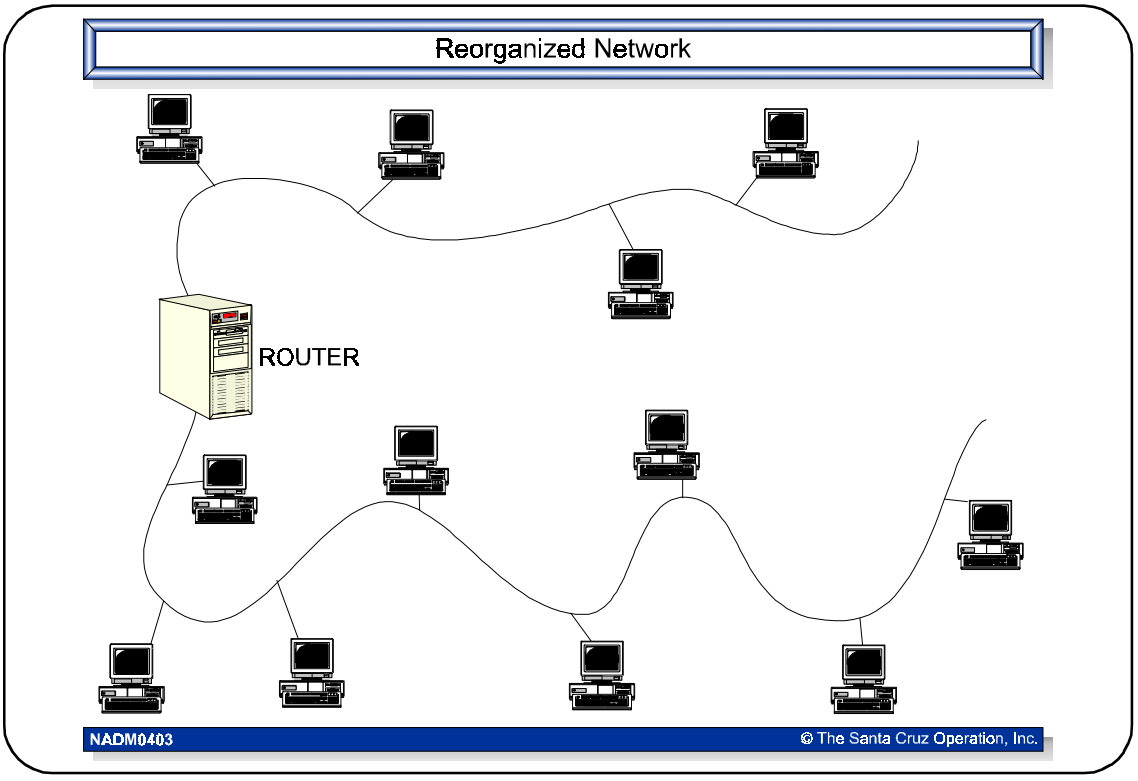
- IP address management can be distributed to local administrators.
- The network topology is hidden from the outside world; using subnets requires only a single route to external gateways.

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


A Single Overloaded Segment
NADM0402
© The Santa Cruz Operation, Inc.

AV 4-3


Reorganized Network
ROUTER
NADM0403
© The Santa Cruz Operation, Inc.

4.1.1 Increasing the number of network nodes

As the number of nodes on a physical network increases, expansion is 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 2-54.

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 4-8.

AV 4-2: A Single Overloaded Segment
Use this slide to show how LANs sometimes grow unchecked.

AV 4-3: Reorganized Network
Use this slide to demonstrate how to resolve the problems shown in the previous slide.

Lead a discussion on what kinds of routers (dedicated, host, etc.) would be useful to deploy in this circumstance. Discuss how to decide which machines go on which subnet.

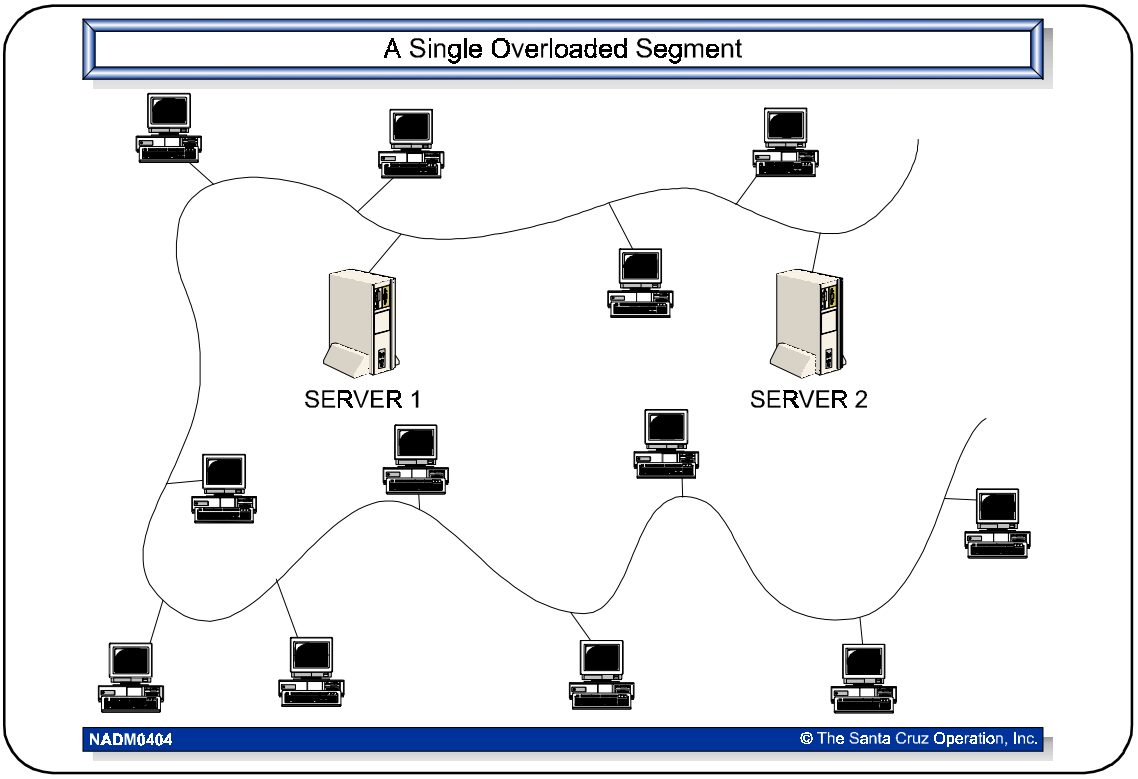
4.1.2 Joining multiple LANs

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.

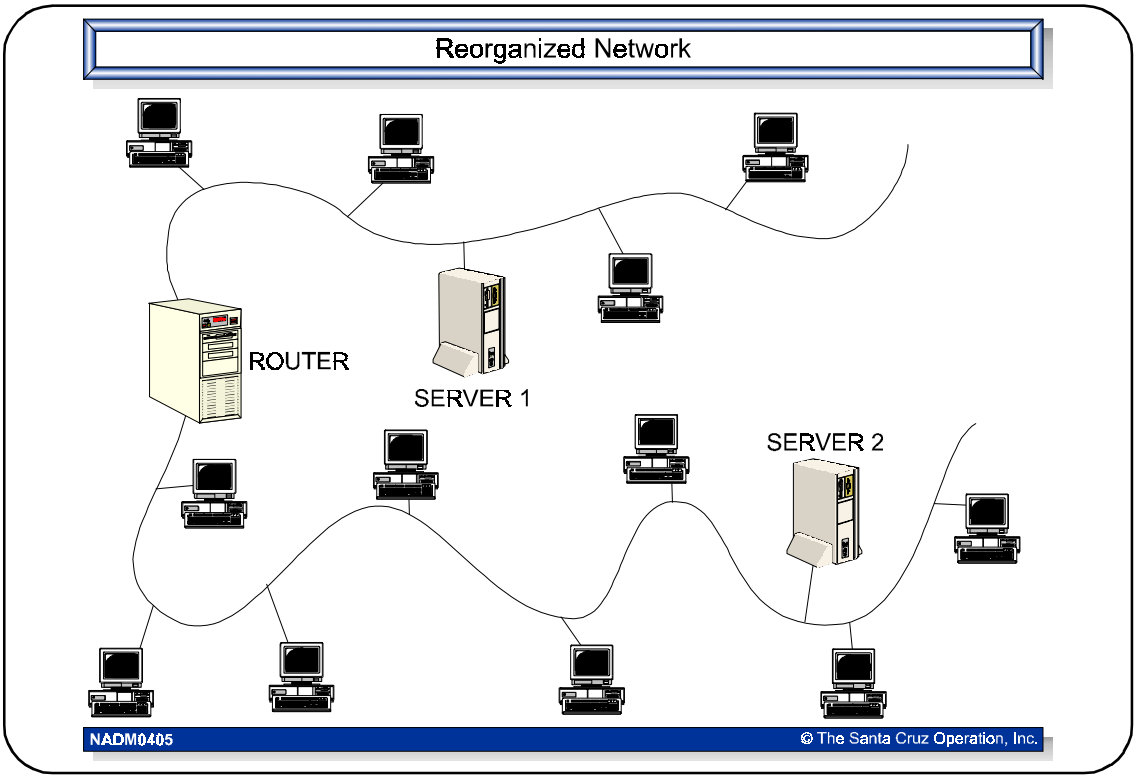
Joining all of your separate LANs into a single enterprise-wide network is the ideal solution. The problems you will face are not entirely different than when the number of nodes increases. Additionally, spanning geographic locations is usually not feasible using a LAN technology (for example, Ethernet).

Subnets provide a method of joining together separate LANs.

AV 4-4


A Single Overloaded Segment
SERVER 1
SERVER 2
NADM0404
© The Santa Cruz Operation, Inc.

AV 4-5


Reorganized Network
ROUTER
SERVER 1
SERVER 2
NADM0405
© The Santa Cruz Operation, Inc.

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.

In other words, if you move a lot of PCs running PC-NFS® and their NFS® server to their own subnet, but the majority of some of the nodes' traffic was from an X Window System application running on a server on the other subnet, this can result in decreased performance for the PCs in question because all those X packets now need to go through a router.

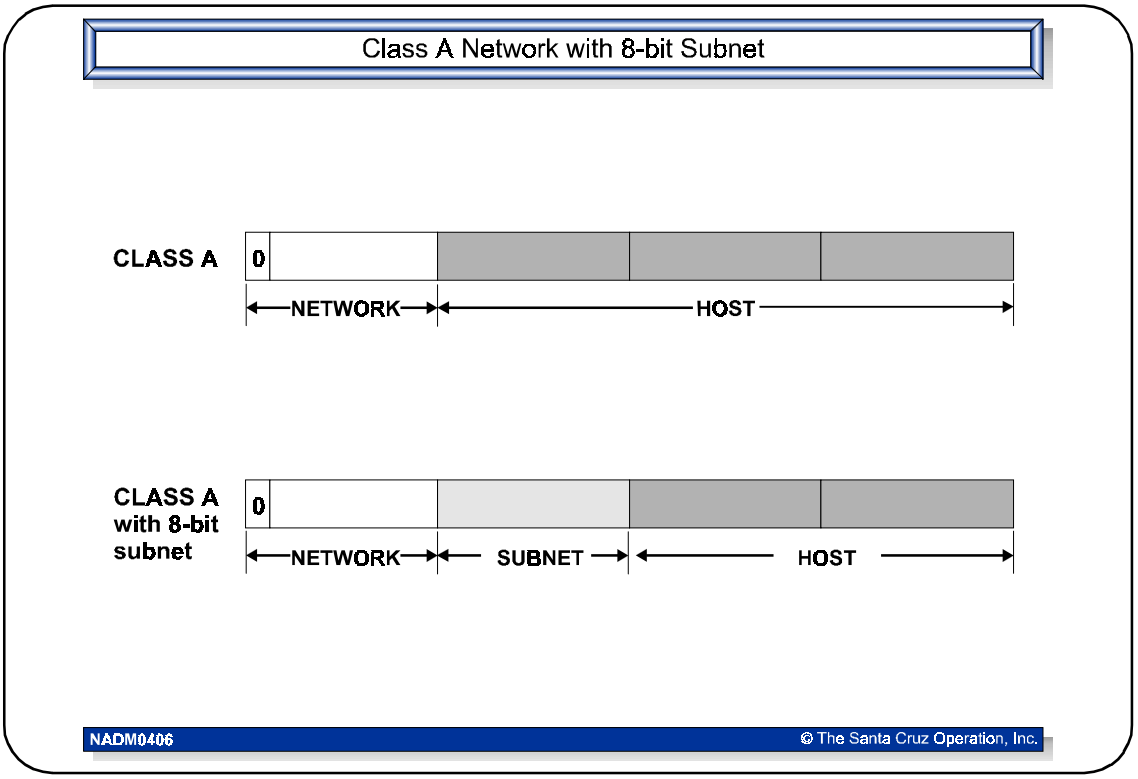
3. Isolate those machines onto their own subnet.

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.

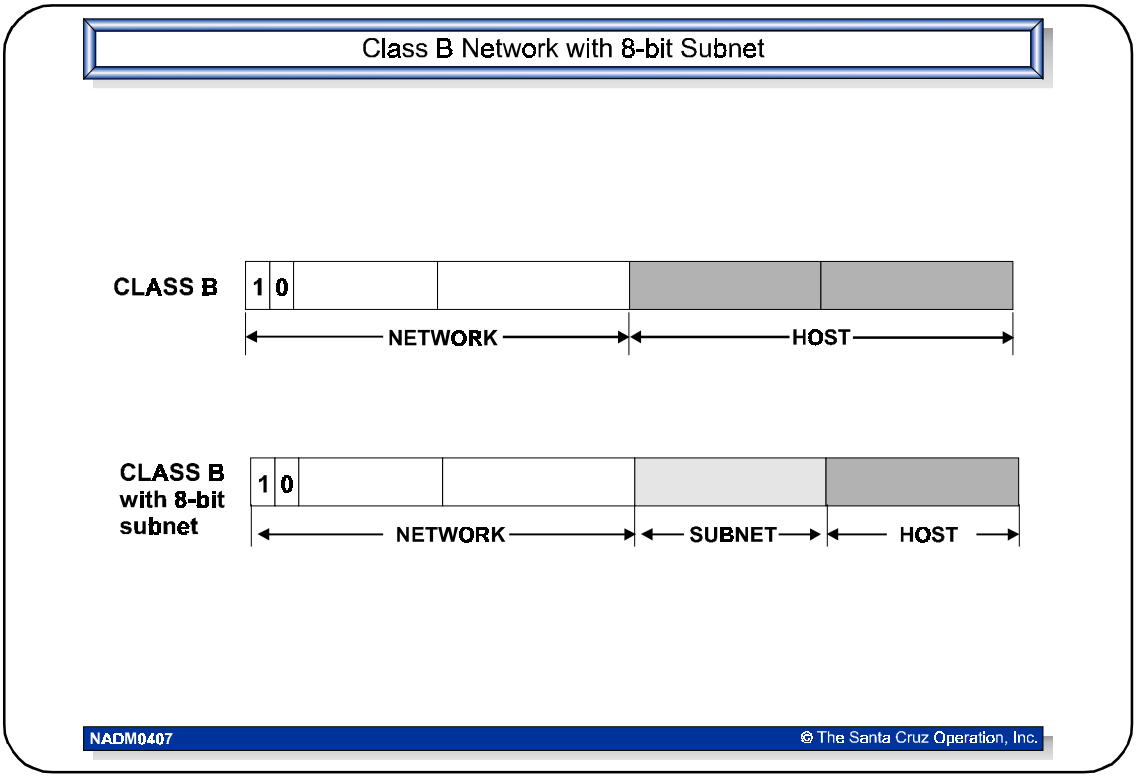
AV 4-4: A Single Overloaded Segment
Point out that this slide is the same as AV 4-2 on page 4-8, except that there are also two servers on the network. Use this to indicate that a network can actually be experiencing both problems at once.

AV 4-5: Reorganized Network
This reorganization solves both problems. Point out the relocation of *server2* to the subnet where most of its requests are generated.

AV 4-6


Class A Network with 8-bit Subnet
CLASS A
0
NETWORK
HOST
CLASS A with 8-bit subnet
0
NETWORK
SUBNET
HOST
NADM0406
© The Santa Cruz Operation, Inc.

AV 4-7


Class B Network with 8-bit Subnet
CLASS B
1
0
NETWORK
HOST
CLASS B with 8-bit subnet
1
0
NETWORK
SUBNET
HOST
NADM0407
© The Santa Cruz Operation, Inc.

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*(SFF) 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.This is 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:

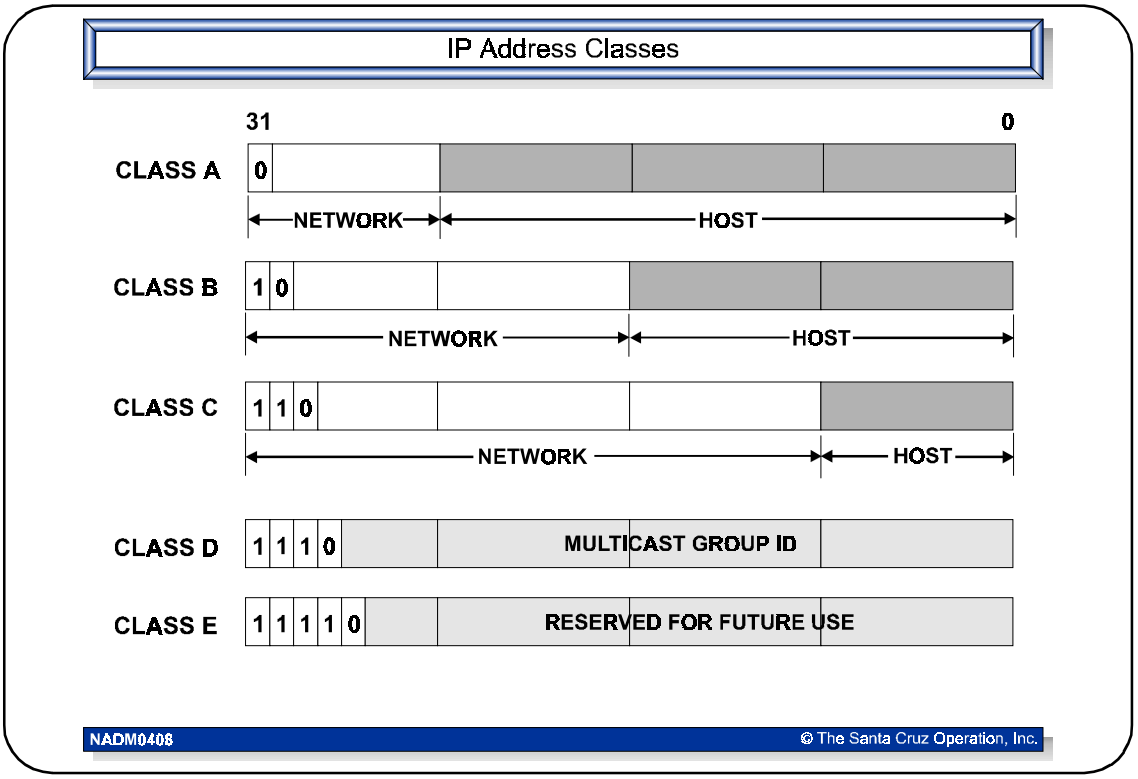
- The number of nodes on your network
- The number of subnets needed
- Future growth

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 4-6: Class A Network with an 8-bit Subnet
This slide is a representation of a Class A network address which uses 8 host bits as a subnet address.

AV 4-7: Class B Network with an 8-bit Subnet
This slide is a representation of a Class B network address which uses 8 host bits as a subnet address.

AV 4-8


IP Address Classes
31
0
CLASS A
0
NETWORK
HOST
CLASS B
1 0
NETWORK
HOST
CLASS C
1 1 0
NETWORK
HOST
CLASS D
1 1 1 0
MULTICAST GROUP ID
CLASS E
1 1 1 1 0
RESERVED FOR FUTURE USE
NADM0408
© The Santa Cruz Operation, Inc.

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 determines how many bits of the IP address are used 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 2: 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

AV 4-8: IP Address Classes
This slide is repeated from Lesson 2 as a reminder of how IP addresses are organized by classes.

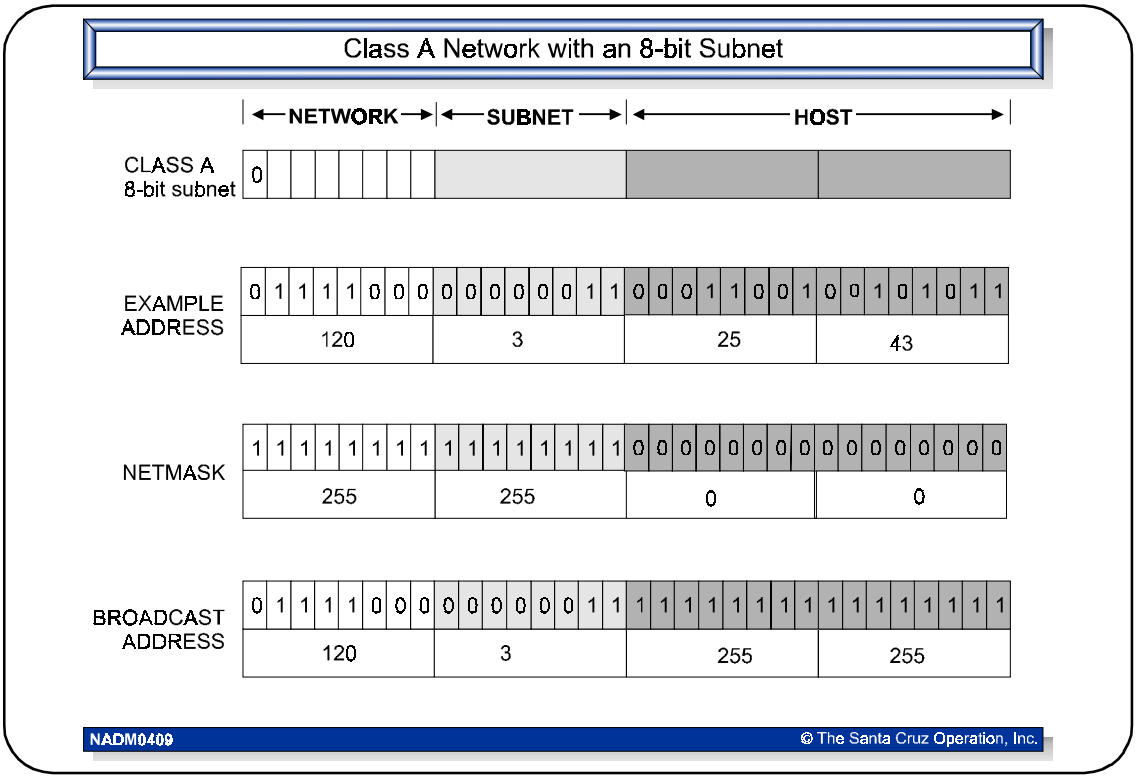
Also recall how the number of network and host bits determine the number of networks available and the number of nodes (hosts) on each network.

Table 3: 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.

AV 4-9


Class A Network with an 8-bit Subnet
NETWORK
SUBNET
HOST
CLASS A
8-bit subnet
0
EXAMPLE
ADDRESS
0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 0 1 1
120
3
25
43
NETMASK
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
255
255
0
0
BROADCAST
ADDRESS
0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
120
3
255
255
NADM0409
© The Santa Cruz Operation, Inc.

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.

You cannot use a one bit subnet, as the only possible values would be 0 or 1, both of which are invalid (all zeros or all ones). Likewise, you cannot use a 23 bit subnet, as only one bit would be left for the host.

4.2.2.1 Class A 8-bit subnets

AV 4-9: Class A Network with an 8-bit Subnet
This slide shows how to calculate the netmask and broadcast address for an example IP address on a Class A network with an 8-bit subnet.

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. The following procedure can be used with AV 4-9 to find the netmask and broadcast address for a node with IP address 120.3.25.43:

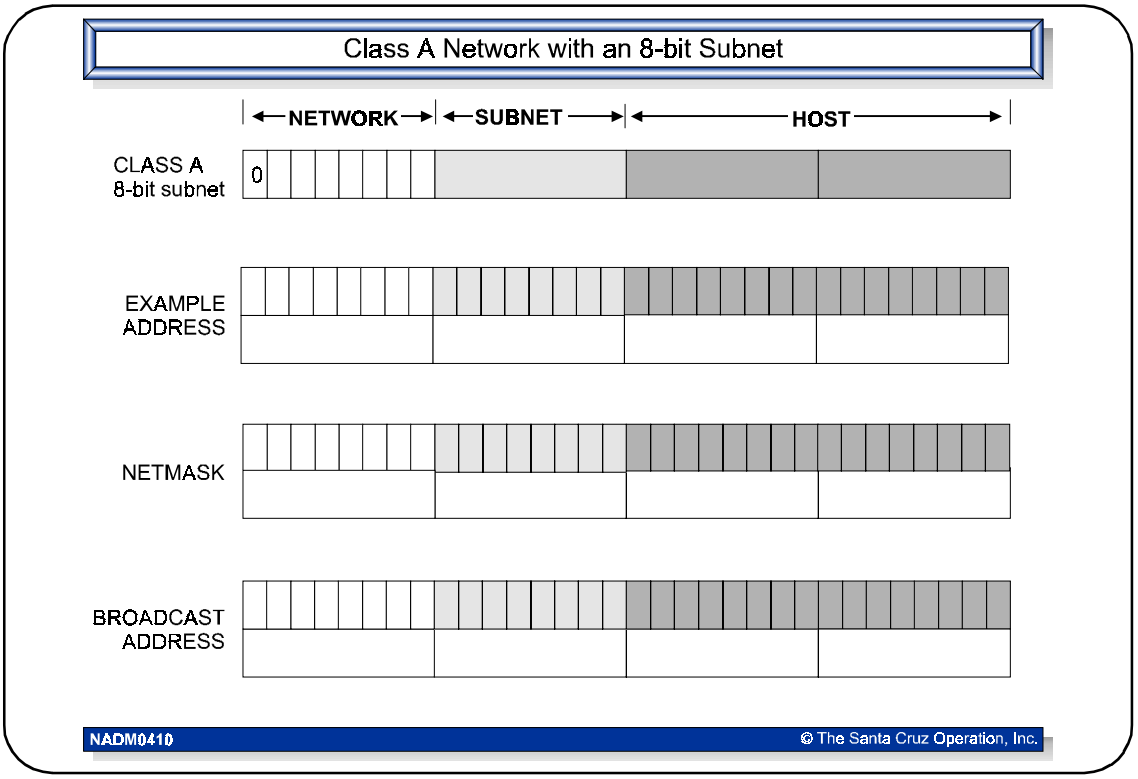
Step 1. Enter the IP address in decimal notation: 120.3.25.43.

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


Class A Network with an 8-bit Subnet
NETWORK
SUBNET
HOST
CLASS A
8-bit subnet
0
EXAMPLE
ADDRESS
NETMASK
BROADCAST
ADDRESS
NADM0410
© The Santa Cruz Operation, Inc.

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

AV 4-10: Class A Network with an 8-bit Subnet
This slide provides a form for students to use with this exercise.

Exercise: Creating Class A 8-bit subnets

Use the form in AV 4-10 on page 4-18 to determine the netmask and broadcast address for IP address 50.75.128.125, assuming an 8-bit subnet.

Address: 00110010 01001011 10000000 01111101

Netmask: 11111111 11111111 00000000 00000000

255.255.0.0

Broadcast: 00110010 01001011 11111111 11111111

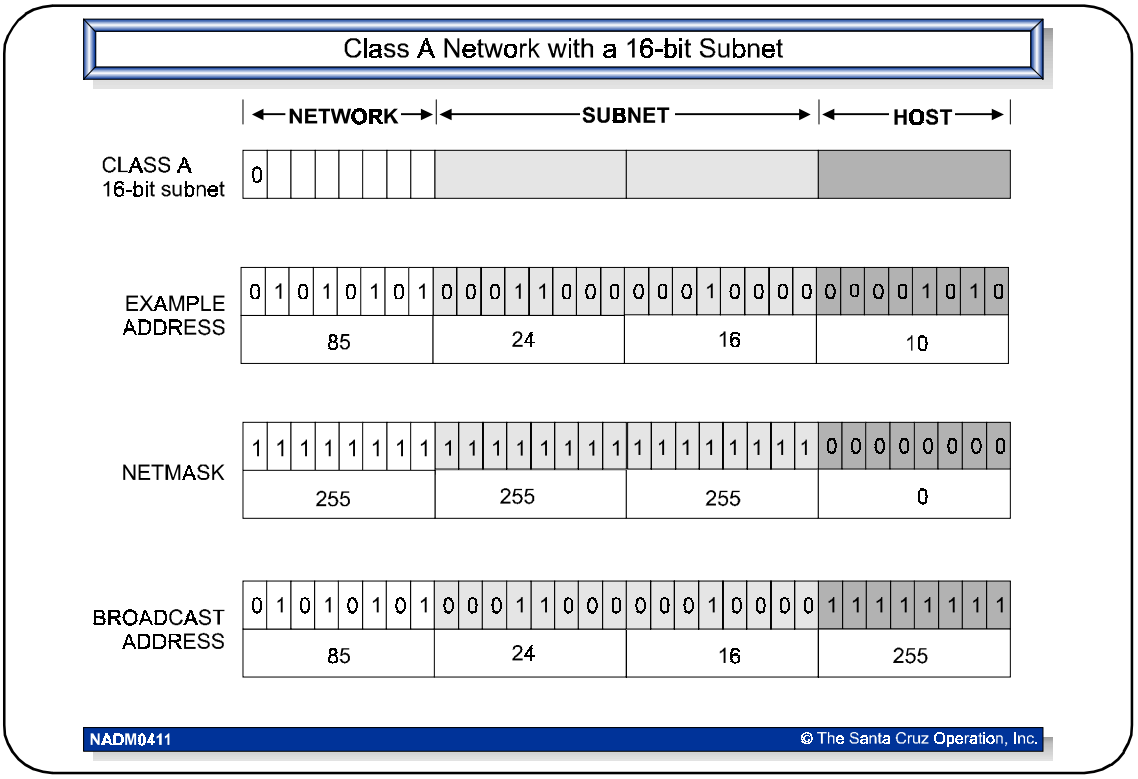
50.75.255.255

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
$
```

The procedure above is repeated several times in the following sections and students are given similar exercises. The repetition is intended as a means of making this process straightforward and uncomplicated. The examples start with the most simple scenarios to build students' confidence in preparation for the more complicated scenarios later.

AV 4-11


Class A Network with a 16-bit Subnet
NETWORK
SUBNET
HOST
CLASS A
16-bit subnet
EXAMPLE
ADDRESS
85
24
16
10
NETMASK
255
255
255
0
BROADCAST
ADDRESS
85
24
16
255
NADM0411
© The Santa Cruz Operation, Inc.

4.2.2.2 Class A 16-bit subnets

AV 4-11: Class A Network with a 16-bit Subnet
This slide shows how to calculate the netmask and broadcast address for an example IP address on a Class A network with a 16-bit subnet.

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`

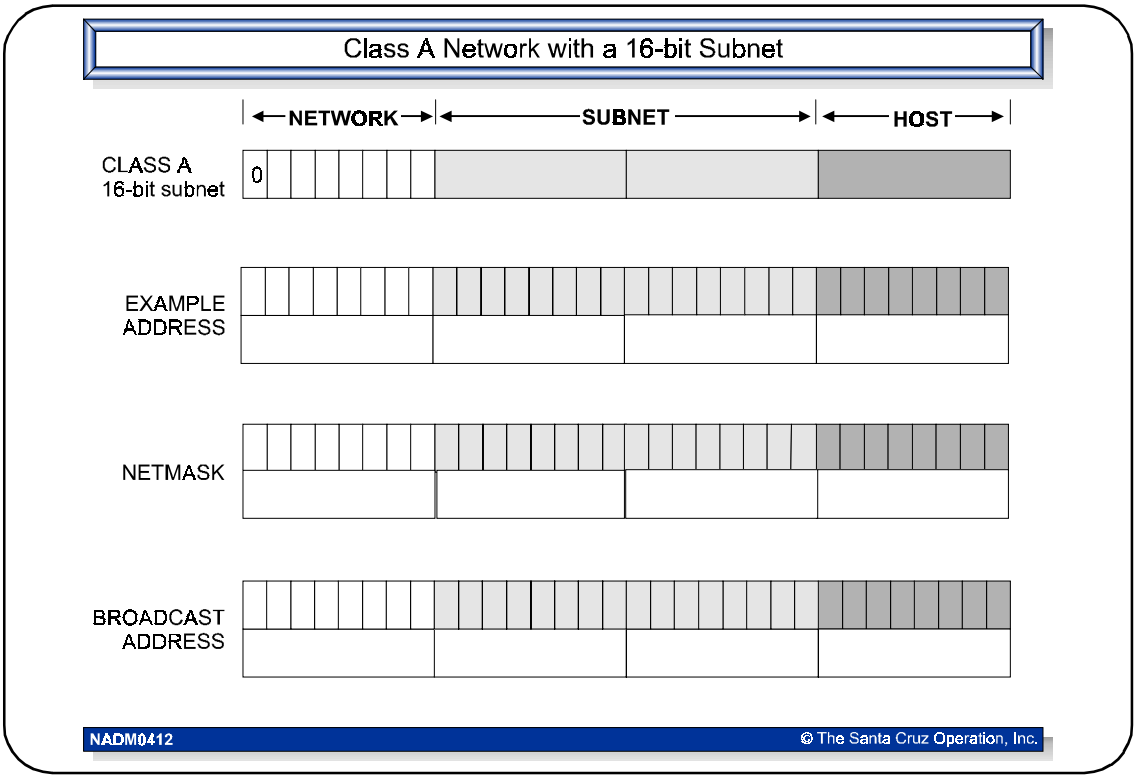
Step 3. Enter the netmask in binary— enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
`11111111 11111111 11111111 00000000`

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


Class A Network with a 16-bit Subnet
NETWORK
SUBNET
HOST
CLASS A
16-bit subnet
0
EXAMPLE
ADDRESS
NETMASK
BROADCAST
ADDRESS
NADM0412
© The Santa Cruz Operation, Inc.

AV 4-12: Class A Network with a 16-bit Subnet
This slide provides a form for students to use with this exercise.

Exercise: Creating Class A 16-bit subnets

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

Address: 01111000 01101110 01111101 00110100

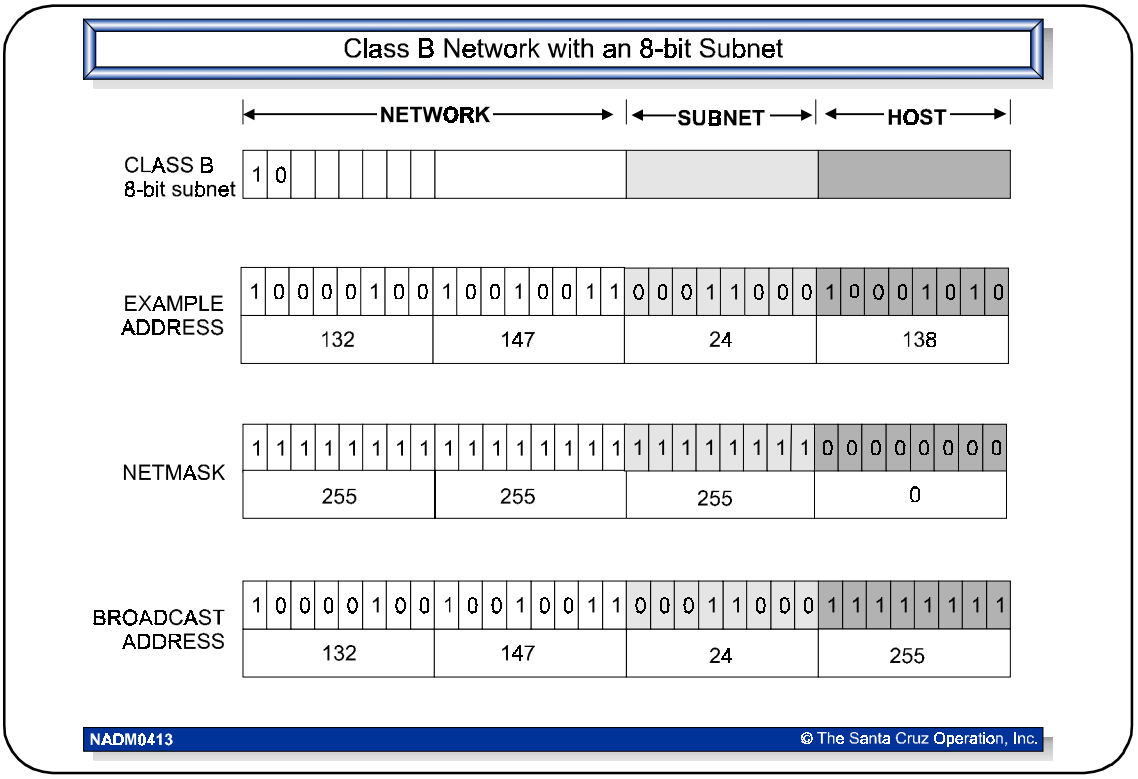
Netmask: 11111111 11111111 11111111 00000000

255.255.255.0

Broadcast: 01111000 01101110 01111101 11111111

120.110.125.255

AV 4-13


Class B Network with an 8-bit Subnet
NETWORK
SUBNET
HOST
CLASS B
8-bit subnet
1 0
EXAMPLE
ADDRESS
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0 1 0 1 0
132
147
24
138
NETMASK
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
255
255
255
0
BROADCAST
ADDRESS
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 1
132
147
24
255
NADM0413
© The Santa Cruz Operation, Inc.

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

AV 4-13: Class B Network with an 8-bit Subnet
This slide shows how to calculate the netmask and broadcast address for an example IP address on a Class A network with a 16-bit subnet.

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.

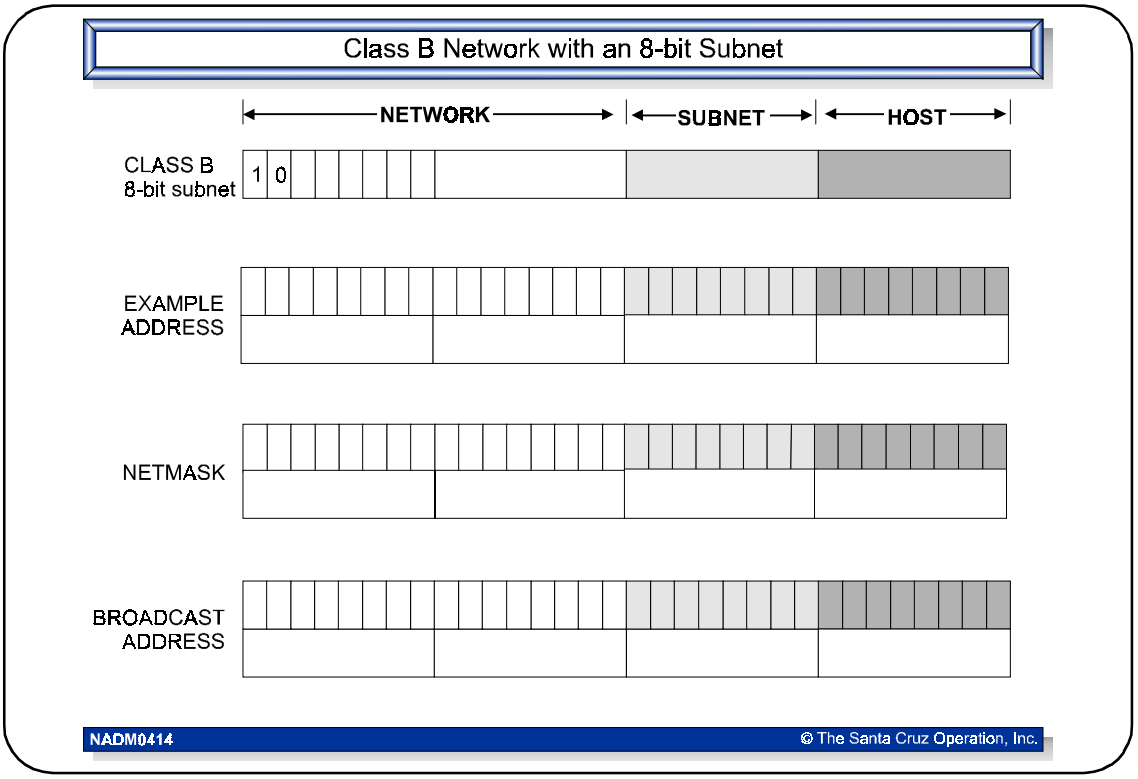
SCO is an example of an organization with a Class B address, 132.147.0.0. SCO has a large number of hosts separated by geography.

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 4-21, you can determine the netmask and broadcast values for example IP address 132.147.24.138, shown in AV 4-13.

AV 4-14


Class B Network with an 8-bit Subnet
NETWORK
SUBNET
HOST
CLASS B
8-bit subnet
1
0
EXAMPLE
ADDRESS
NETMASK
BROADCAST
ADDRESS
NADM0414
© The Santa Cruz Operation, Inc.

AV 4-14: Class B Network with an 8-bit Subnet
This slide provides a form for students to use with this exercise.

Exercise: Creating Class B 8-bit subnets

Use the form in AV 4-14 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.

Address: 10000100 10010011 10010110 11011100

132.147.150.220

Netmask: 11111111 11111111 11111111 00000000

255.255.255.0

Broadcast: 10000100 10010011 10010110 11111111

132.147.150.255

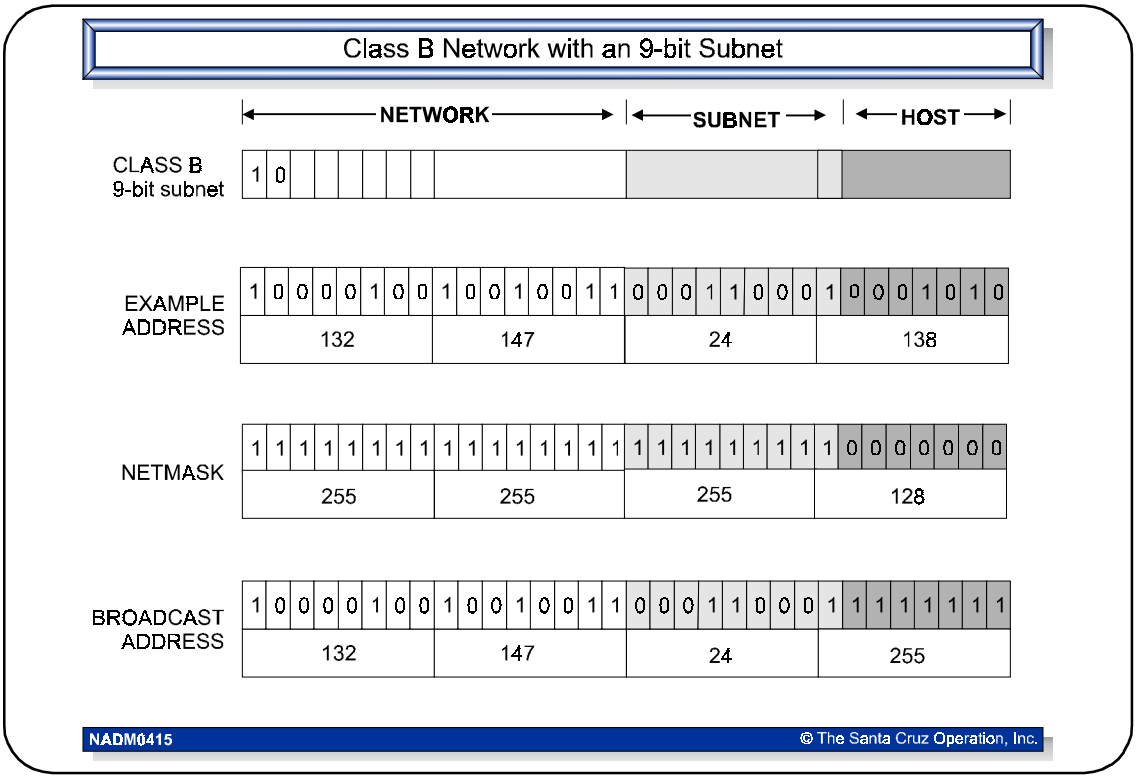
4.2.3.2 Class B non 8-bit subnets

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.

If 9 bits are used 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


Class B Network with an 9-bit Subnet
NETWORK
SUBNET
HOST
CLASS B
9-bit subnet
1 0
EXAMPLE
ADDRESS
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0 1 0 1 0
132
147
24
138
NETMASK
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
255
255
255
128
BROADCAST
ADDRESS
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 1 1 1 1 1 1 1 1
132
147
24
255
NADM0415
© The Santa Cruz Operation, Inc.

AV 4-15: Class B 9-bit Subnet
This slide illustrates how a 9-bit subnet might work better when more subnets with fewer nodes are needed.

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

Use the slide to show how the 9-bit subnet works. In binary, the subnet number is `000110001`, which is decimal 49. In binary, the host number is `0001010`, which is decimal 10.

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`

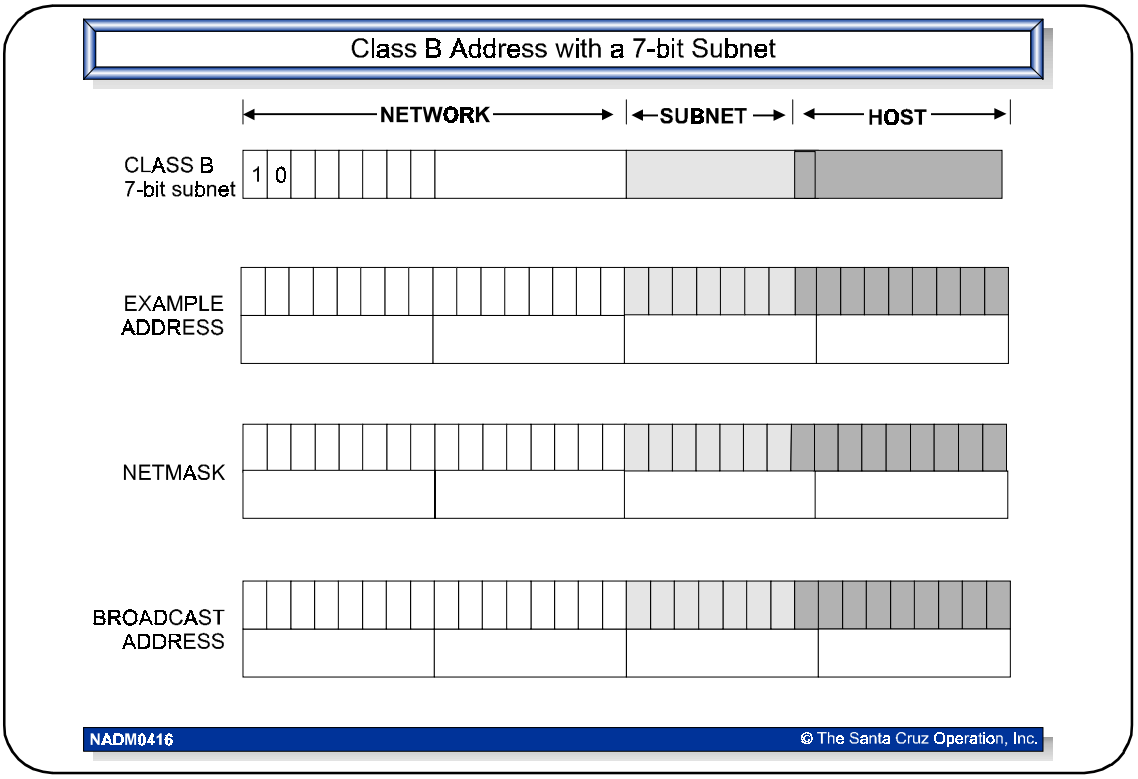
Step 3. Enter the netmask in binary—enter a one (1) for each network and subnet address bit and a zero (0) for each host address bit:
`11111111 11111111 11111111 10000000`

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


Class B Address with a 7-bit Subnet
NETWORK
SUBNET
HOST
CLASS B
7-bit subnet
1
0
EXAMPLE
ADDRESS
NETMASK
BROADCAST
ADDRESS
NADM0416
© The Santa Cruz Operation, Inc.

AV 4-16: Class B 7-bit Subnet
This slide provides a form for students to use with this exercise.

Exercise: Creating Class B 7-bit subnets

Use the form in AV 4-16 on page 4-30 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?

Address: 10101111 11001000 00010101 00001000

Netmask: 11111111 11111111 11111110 00000000

255.255.254.0

Broadcast: 10101111 11001000 00010101 11111111

175.200.21.255

Subnet number: 10 (binary 0001010)

Host number: 264 (binary 1000010000)

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)

You cannot have a 1-bit subnet. The only possible values would be 0 or 1, both of which are invalid (all zeros or all ones). Likewise, you cannot have a 7-bit subnet, because that would leave only one bit for the host.

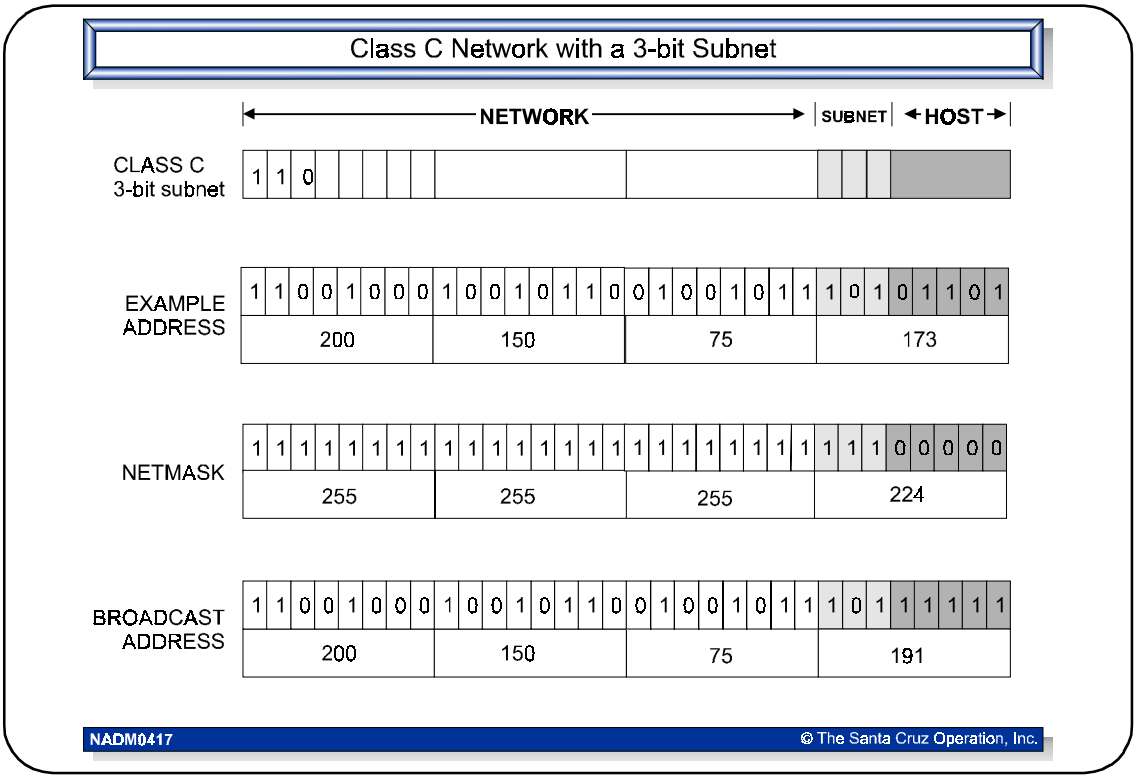
The next example shows how the Widget Company uses its Class C network, 200.150.75.0.

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 addresses:

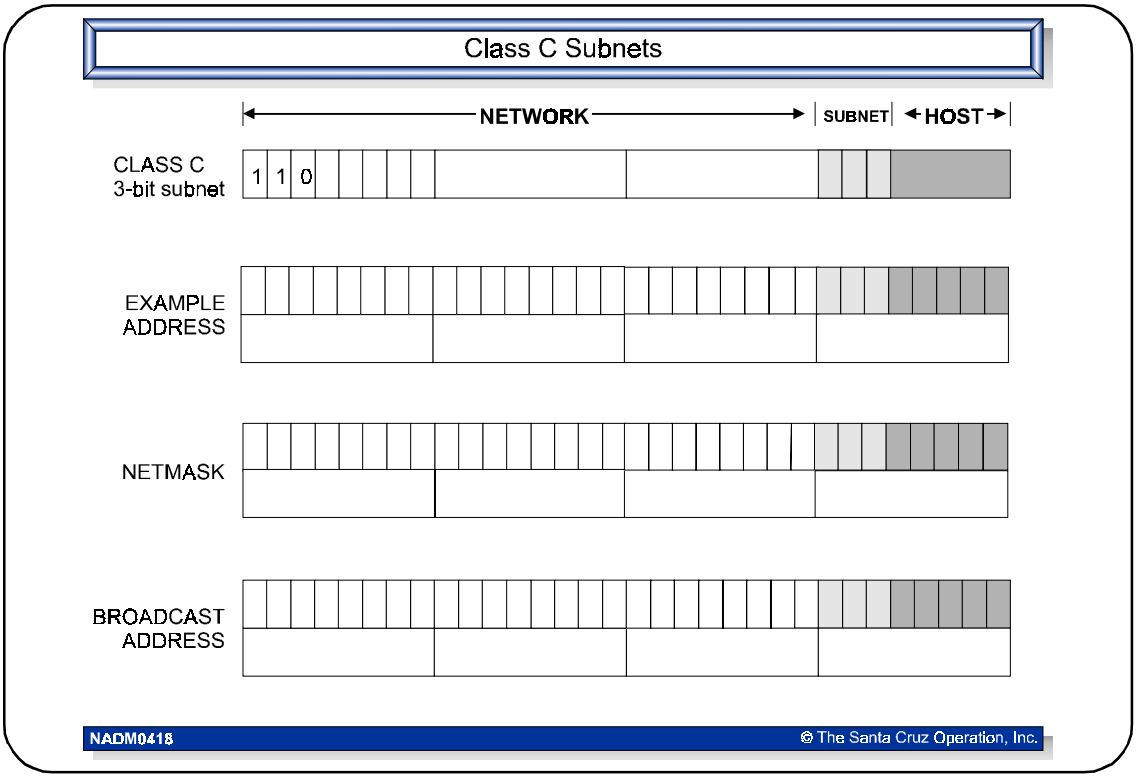
Table 4: Widget Company Subnets

Subnet name	Network number	IP address range for each subnet
Subnet 1 (binary 001)	200.150.75.32	200.150.75.33 - 200.150.75.62
Subnet 2 (binary 010)	200.150.75.64	200.150.75.65 - 200.150.75.94
Subnet 3 (binary 011)	200.150.75.96	200.150.75.97 - 200.150.75.126
Subnet 4 (binary 100)	200.150.75.128	200.150.75.129 - 200.150.75.158
Subnet 5 (binary 101)	200.150.75.160	200.150.75.161 - 200.150.75.190.
Subnet 6 (binary 110)	200.150.75.192	200.150.75.193 - 200.150.75.222

AV 4-17


Class C Network with a 3-bit Subnet
NETWORK
SUBNET
HOST
CLASS C
3-bit subnet
1 1 0
EXAMPLE
ADDRESS
1 1 0 0 1 0 0 0 1 0 0 1 0 1 1 0 0 1 0 0 1 0 1 1 1 0 1 0 1 1 0 1
200
150
75
173
NETMASK
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
255
255
255
224
BROADCAST
ADDRESS
1 1 0 0 1 0 0 0 1 0 0 1 0 1 1 0 0 1 0 0 1 0 1 1 1 0 1 1 1 1 1 1
200
150
75
191
NADM0417
© The Santa Cruz Operation, Inc.

AV 4-18


Class C Subnets
NETWORK
SUBNET
HOST
CLASS C
3-bit subnet
1 1 0
EXAMPLE
ADDRESS
NETMASK
BROADCAST
ADDRESS
NADM0418
© The Santa Cruz Operation, Inc.

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

AV 4-17: Class C 3-bit Subnet
Use this slide to show students how the subnetwork numbers are derived; use the example IP address to determine the netmask and broadcast address.

AV 4-18: Class C Subnets
This slide can be used as an aid for the next exercise asking students to give the host and subnet numbers for IP address 200.150.75.173 using 2-, 3-, and 4-bit subnets. Note that this slide appears on the page opposite the exercise in the Student Guide.

Exercise: Class C subnet addresses

Use the form in AV 4-18 on page 4-34 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

It is host 13 on subnet 5; the binary split is 101 01101:

101 = 5

01101 = 13

- With a 2-bit subnet

It is host 45 on subnet 2; the binary split is 10 101101:

10 = 2

101101 = 45

- With a 4-bit subnet

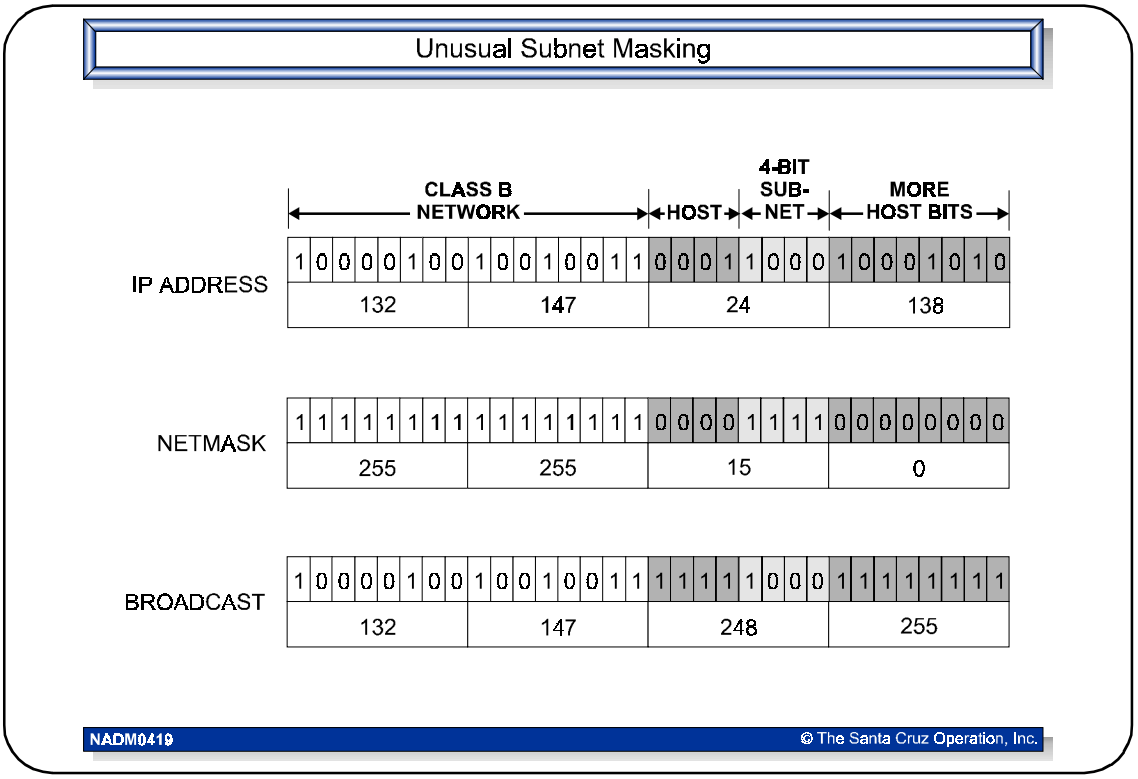
It is host 13 on subnet 10; the binary split is 1010 1101:

1010 = 10

1101 = 13

Instructor Guide

AV 4-19


Unusual Subnet Masking
CLASS B NETWORK
HOST
4-BIT SUB-NET
MORE HOST BITS
IP ADDRESS
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0 1 0 1 0
132
147
24
138
NETMASK
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0
255
255
15
0
BROADCAST
1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1
132
147
248
255
NADM0419
© The Santa Cruz Operation, Inc.

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: Unusual Subnet Masking
This slide shows how to determine the netmask and broadcast values for a Class B subnet with the subnet portion non-contiguous with the network portion. This is the same address as before, but the netmask is different here.

AV 4-19 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.

394 = `0001 1000 1010` (binary)

8 = `1000` (binary)

Point out that it is not a good idea to use non-contiguous subnet masks. The example above was difficult enough to use with a netmask of 255.255.15.0 (`11111111 11111111 00001111 00000000`).

The following netmask, although "valid," is even harder to use:

255.255.170.85 (`11111111 11111111 10101010 01010101`)

With this netmask, the IP address 132.47.24.138 would correspond to host 75 (`01001011`) on subnet 32 (`00100000`)

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 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(C)	Invokes a command-line calculator

Relevant files

/etc/hosts	Local table for hostname to IP address matching. See *hosts*(SFF).

4.5 Answers to Excercises and Review Questions

This section provides suggested answers to exercises and review questions.

Exercise: Creating Class A 8-bit subnets

```
Address:   00110010 01001011 10000000 01111101
Netmask:   11111111 11111111 00000000 00000000
           255.255.0.0
Broadcast: 00110010 01001011 11111111 11111111
           50.75.255.255
```

Exercise: Creating Class A 16-bit subnets

```
Address:   01111000 01101110 01111101 00110100
Netmask:   11111111 11111111 11111111 00000000
           255.255.255.0
Broadcast: 01111000 01101110 01111101 11111111
           120.110.125.255
```

Exercise: Creating Class B 8-bit subnets

```
Address:   10000100 10010011 10010110 11011100
           132.147.150.220
Netmask:   11111111 11111111 11111111 00000000
           255.255.255.0
Broadcast: 10000100 10010011 10010110 11111111
           132.147.150.255
```

Exercise: Creating Class B 7-bit subnets

```
Address:   10101111 11001000 00010101 00001000
Netmask:   11111111 11111111 11111110 00000000
           255.255.254.0
Broadcast: 10101111 11001000 00010101 11111111
           175.200.21.255

Subnet number:  10 (binary 0001010)
Host number:   264 (binary 1000010000)
```

Exercise: Class C subnet addresses

- With a 3-bit subnet:

  It is host 13 on subnet 5; the binary split is `101 01101`:
  `101 = 5`
  `01101 = 13`

- With a 2-bit subnet

  It is host 45 on subnet 2; the binary split is `10 101101`:
  `10 = 2`
  `101101 = 45`

- With a 4-bit subnet

  It is host 13 on subnet 10; the binary split is `1010 1101`:
  `1010 = 10`
  `1101 = 13`

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
   - 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

Instructor Guide

Lesson 5

Routing

Suggested lesson duration

This lesson should take approximately 2.5 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

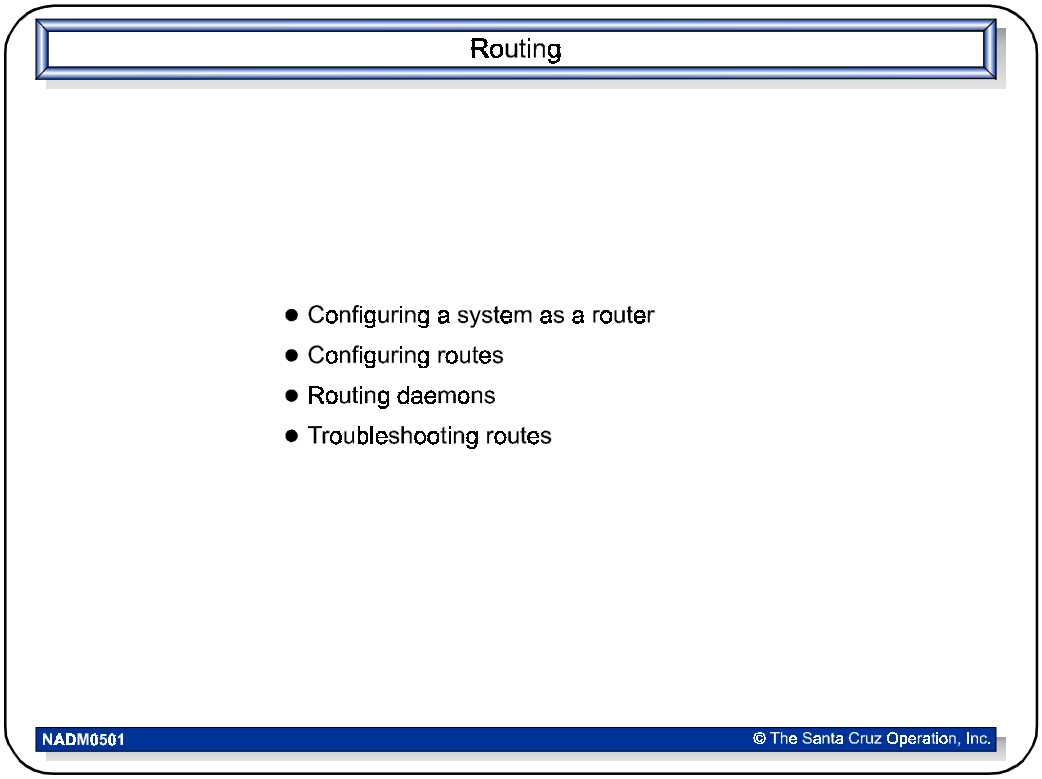
Set up the instructor's machine with two (2) network adapters to demonstrate routing. Also configure **gated**(ADMN) on the instructor's machine to run the Open Shortest Path First (OSPF) protocol.

Instructor preparation

Review the online documentation on IP routing. Further, **gated**(ADMN) is discussed from the "when do you use it" point of view. You should know how to configure **gated** should students want to do this.

Be sure to have the third edition of Comer, Douglas E.***, Internetworking with TCP/IP, Volume I: Principles, Protocols, and Architecture***, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1995, ISBN: 0-13-216987-8, readily available.

AV 5-1


Routing
Configuring a system as a router
Configuring routes
Routing daemons
Troubleshooting routes
NADM0501
© The Santa Cruz Operation, Inc.

Introduction

A router connects two networks together. You can connect networks to control network traffic or to 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 need to add additional network adapters to your system and then configure the software. This lesson describes how to do this.

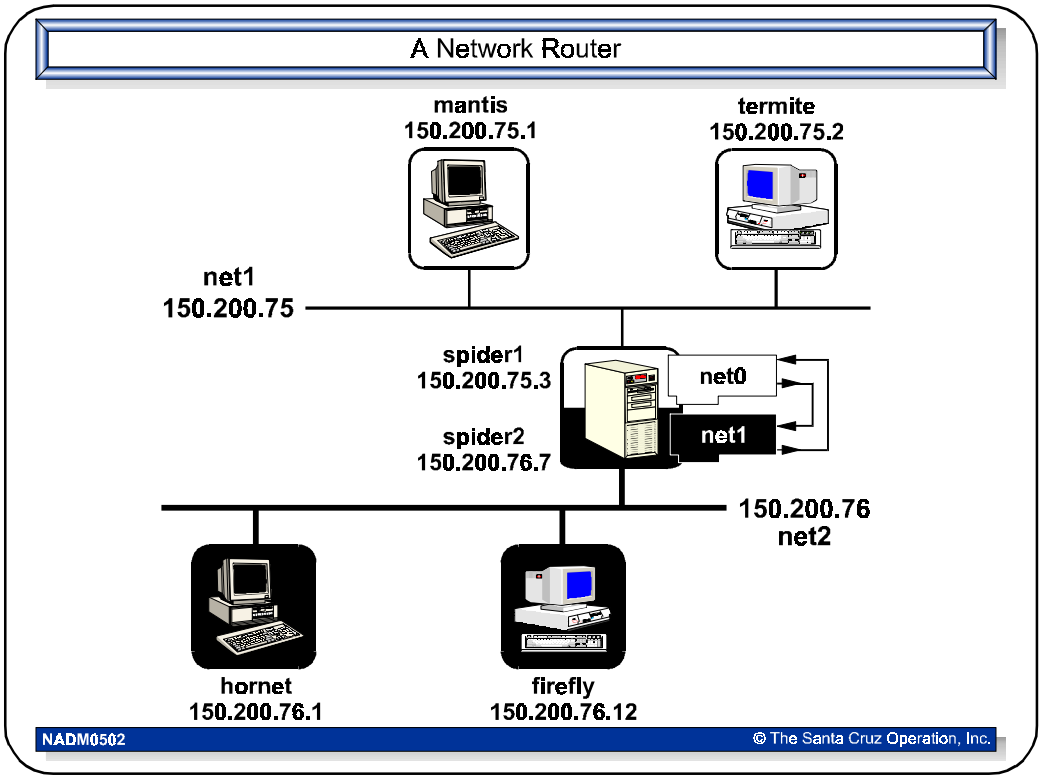
Objectives

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 5-1: Routing
The first slide describes the overall topics covered in this lesson.

AV 5-2


A Network Router
mantis
150.200.75.1
termite
150.200.75.2
net1
150.200.75
spider1
150.200.75.3
spider2
150.200.76.7
net0
net1
150.200.76
net2
hornet
150.200.76.1
firefly
150.200.76.12
NADM0502
© The Santa Cruz Operation, Inc.

5.1 Configuring a System as a Router

Configuring your system to be a router has two basic parts:

- Installing additional network adapters
- Running the **Network Configuration Manager** to configure the software

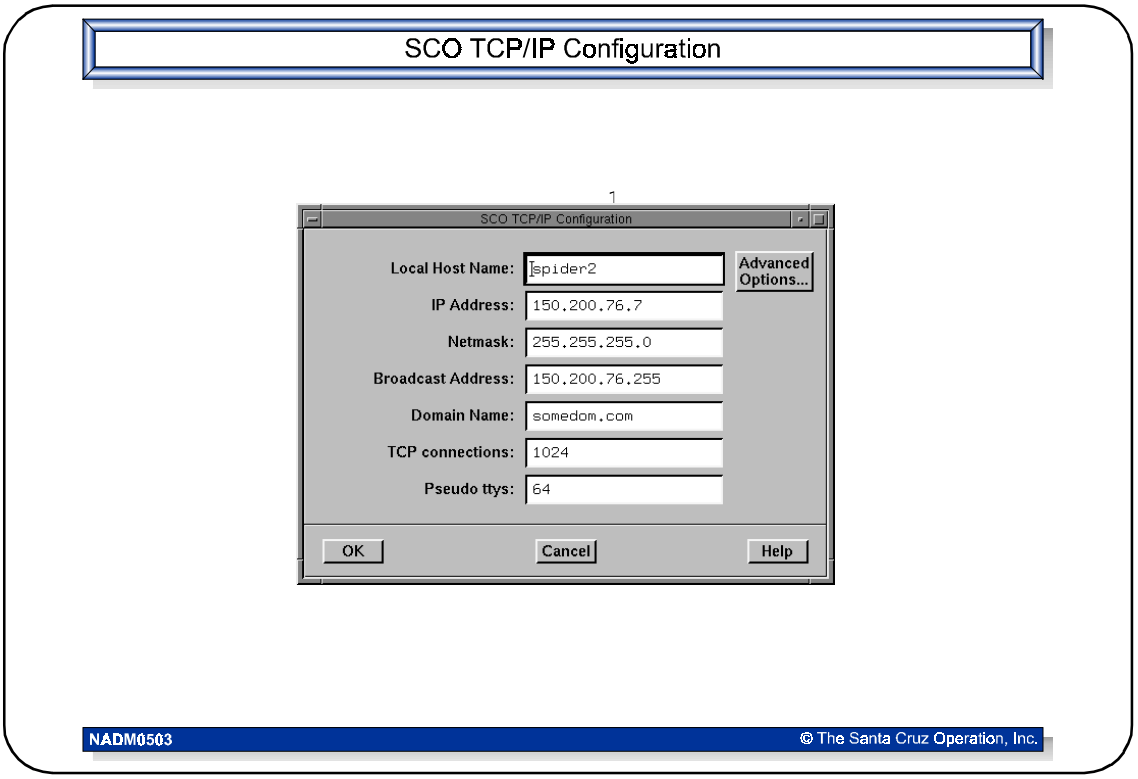
Routing between your network and others then happens automatically.

Each of these parts are discussed in the following sections.

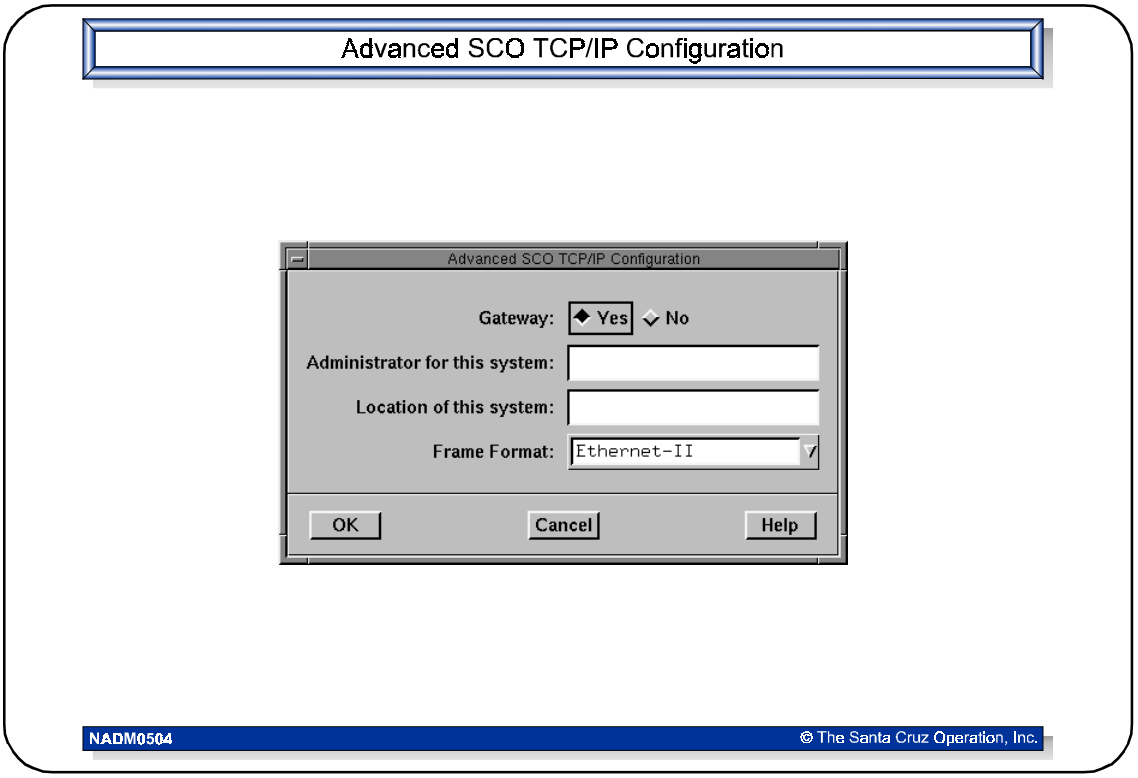
AV 5-2: A Network Router
This slide shows *spider* as having two network adapters (*net0* and *net1*), each on a different network (and with two different IP addresses and names: *spider1* and *spider2*). When `ipforwarding` is set to `1` (see Section 5.1.2 on page 5-11), IP packets can be routed from one of these networks to another.

Note that it is common to use the same base system name for each interface in a router. For example, *spider1* and *spider2* rather than *cricket* and *bee*.

AV 5-3


SCO TCP/IP Configuration
SCO TCP/IP Configuration
Local Host Name:
spider2
Advanced Options...
IP Address:
150.200.76.7
Netmask:
255.255.255.0
Broadcast Address:
150.200.76.255
Domain Name:
somedom.com
TCP connections:
1024
Pseudo ttys:
64
OK
Cancel
Help
NADM0503
© The Santa Cruz Operation, Inc.

AV 5-4


Advanced SCO TCP/IP Configuration
Advanced SCO TCP/IP Configuration
Gateway:
Yes
No
Administrator for this system:
Location of this system:
Frame Format:
Ethernet-II
OK
Cancel
Help
NADM0504
© The Santa Cruz Operation, Inc.

5.1.1 Installing additional network adapters

To configure a second network adapter, using the **Network Configuration Manager**, follow these steps:

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

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

During the configuration, you will provide the same kind of information you did to configure previous network adapters, but you will 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. A window displays that is similar to the one shown in AV 5-3.

Step 5. Click on **Advanced Options** in the **SCO TCP/IP Configuration** window. (See AV 5-4).

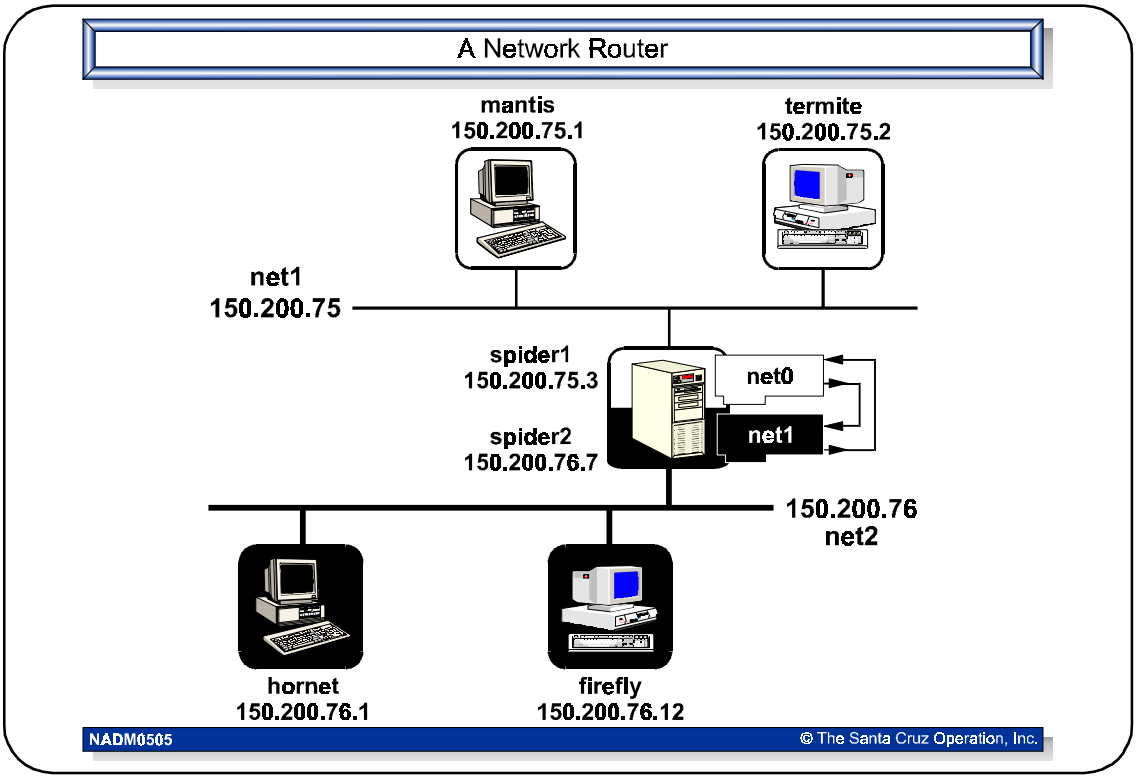
Step 6. Verify that "Gateway" is set to **Yes**.

Step 7. Click on **OK** in the **Advanced SCO TCP/IP Configuration** window.

AV 5-3: SCO TCP/IP Configuration
Use this slide to indicate the **Advanced Options** button.

AV 5-4: Advanced SCO TCP/IP Configuration
This slide shows a sample screen shot of the **Advanced SCO TCP/IP Configuration** window. It is accessed from the **SCO TCP/IP Configuration** window by clicking on the **Advanced Options** button.

AV 5-5


A Network Router
mantis
150.200.75.1
termite
150.200.75.2
net1
150.200.75
spider1
150.200.75.3
spider2
150.200.76.7
net0
net1
150.200.76
net2
hornet
150.200.76.1
firefly
150.200.76.12
NADM0505
© The Santa Cruz Operation, Inc.

5.1.2 Kernel parameters

When you use the **Network Configuration Manager** to configure your SCO OpenServer Release 5 system as a router, kernel parameters are set. Two of these parameters are `ipforwarding` and `ipsendredirects`.

5.1.2.1 The `ipforwarding` parameter

AV 5-5: A Network Router
Use this slide to illustrate the points below.

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 you set the parameter for one adapter, you do not need 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`).

5.1.2.2 The `ipsendredirects` parameter

Another kernel parameter set by netconfig is `ipsendredirects`. When this parameter is set to one (1), the router uses Internet Control Message Protocol (ICMP) packets to communicate changes in routing information.

For example, if a system sends a packet to a router which knows of a more direct route, the router sends an ICMP redirect message to the system. The system creates a new, dynamic route based on the ICMP redirect.

When set to zero (0), the system does not use ICMP packets to communicate changes in routing formation.

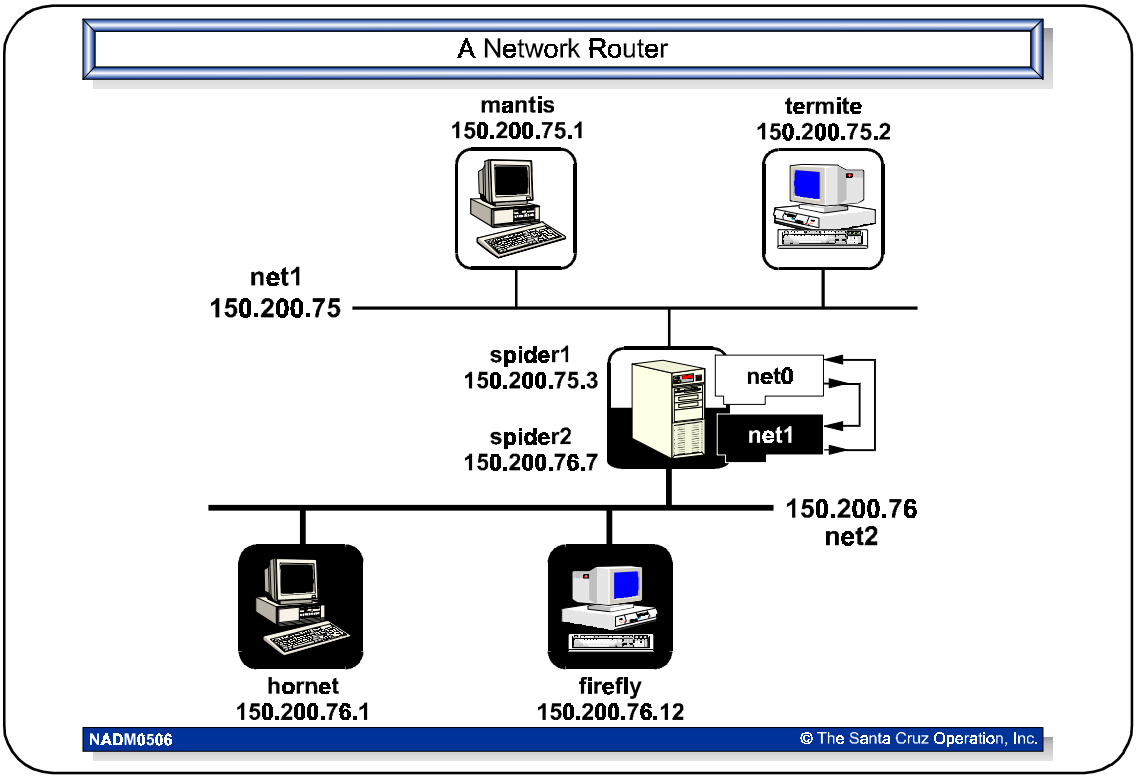
5.1.2.3 Modifying kernel parameters

The **inconfig**(ADMN) command can also be used to configure the `ipforwarding` and `ipsendredirects` parameters. The **inconfig**(ADMN) command modifies */etc/default/inet*. Warning: This file should *not* be edited manually.

To enable routing, enter:

```
inconfig ipforwarding 1
inconfig ipsendredirects 1
```

AV 5-6


A Network Router
mantis
150.200.75.1
termite
150.200.75.2
net1
150.200.75
spider1
150.200.75.3
spider2
150.200.76.7
net0
net1
150.200.76
net2
hornet
150.200.76.1
firefly
150.200.76.12
NADM0506
© The Santa Cruz Operation, Inc.

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

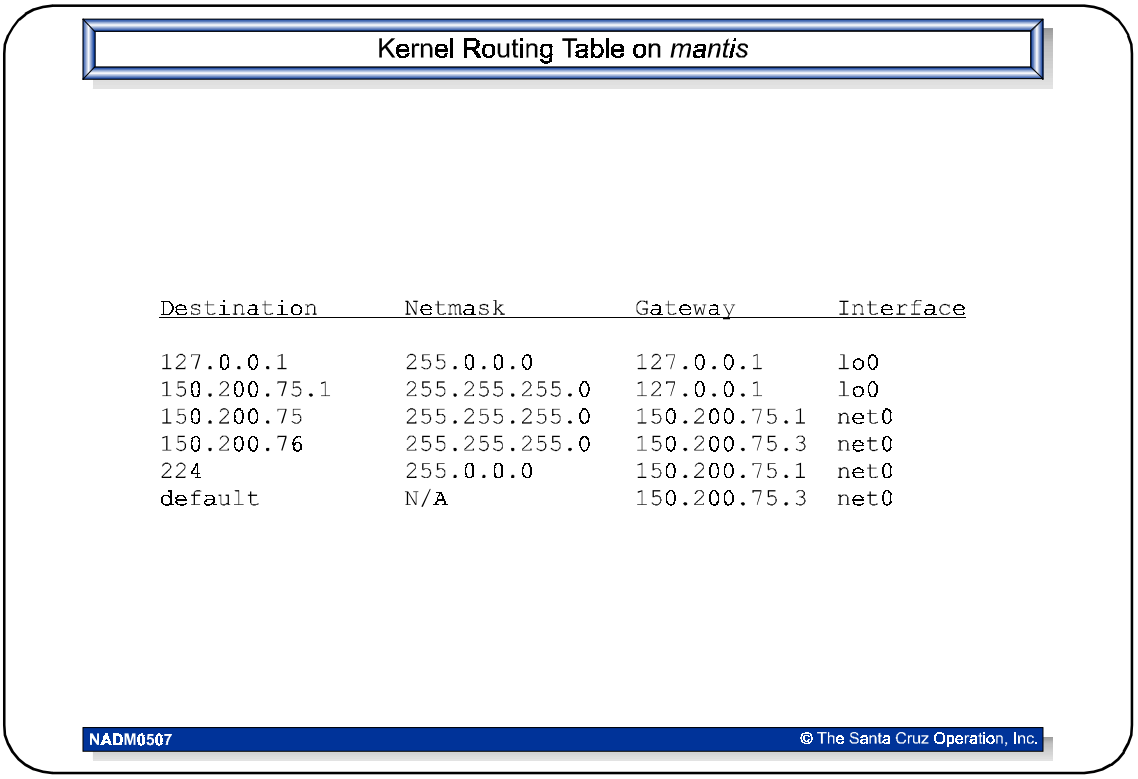
AV 5-6: A Network Router
Use this slide to illustrate when direct and indirect routes are used.

There are two types of routing:

direct — 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 *hornet* and *firefly*.

indirect — 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 *hornet* and *mantis* (through the *spider1* and *spider2* interfaces on the router).

AV 5-7


Kernel Routing Table on *mantis*
Destination Netmask Gateway Interface
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
NADM0507
© The Santa Cruz Operation, Inc.

5.2.2 The routing table

AV 5-7: Kernel Routing Table on *mantis*
Use this slide (and the previous slide) to illustrate what some of the entries in the kernel routing table on *mantis* look like.

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

- To the host 127.0.0.1 (*localhost*) through the *lo0* loopback interface
- 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* interface.
- 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.

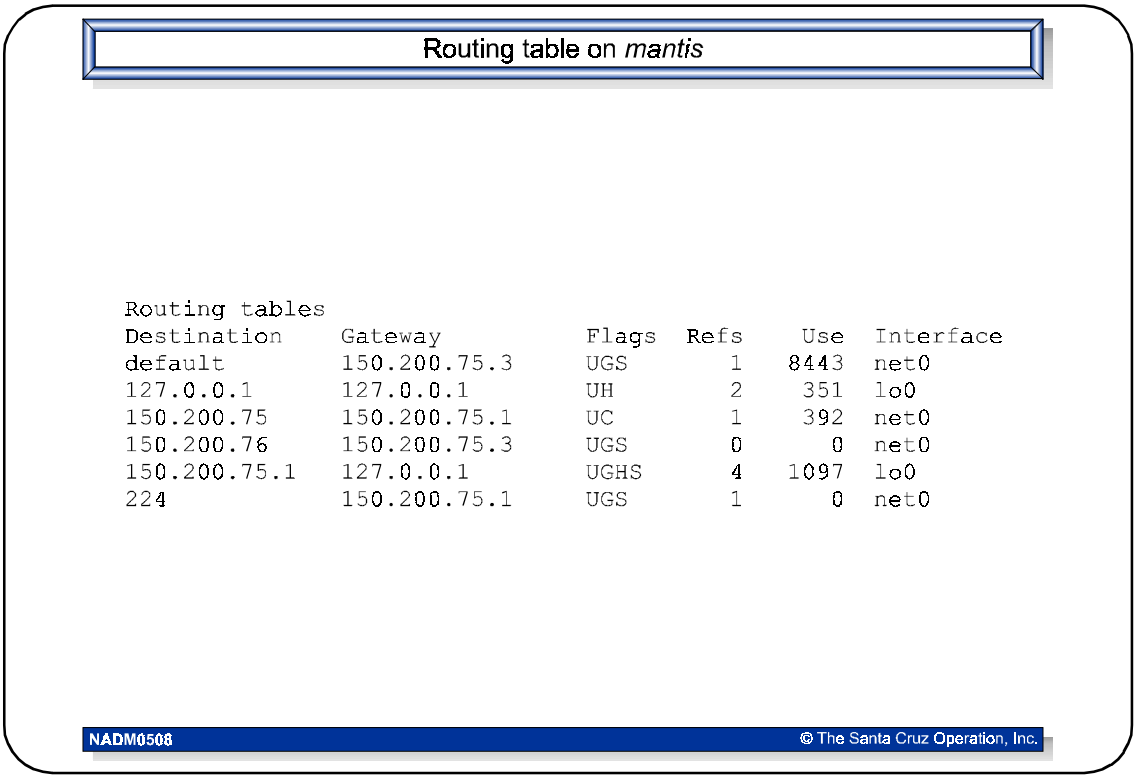
Ask students how many routes are created for network adapters on the router and what they would be.

- To a multicast network

Some of the entries that are in the kernel routing table for *mantis* (in the previous example) are shown in AV 5-7. 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


Routing table on *mantis*
Routing tables
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
150.200.75 150.200.75.1 UC 1 392 net0
150.200.76 150.200.75.3 UGS 0 0 net0
150.200.75.1 127.0.0.1 UGHS 4 1097 lo0
224 150.200.75.1 UGS 1 0 net0
NADM0508
© The Santa Cruz Operation, Inc.

5.2.3 The netstat(ADMN) command

AV 5-8: Routing table on *mantis*
This slide shows sample output from the **netstat** command from *mantis*. Use this slide to discuss the output of the command with students.

Note that, depending on network activity, the field values may differ upon successive invocations.

Use the **netstat**(ADMN) 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**(ADMN) 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.

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(ADMN)

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

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

5.2.4 Updating routes

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

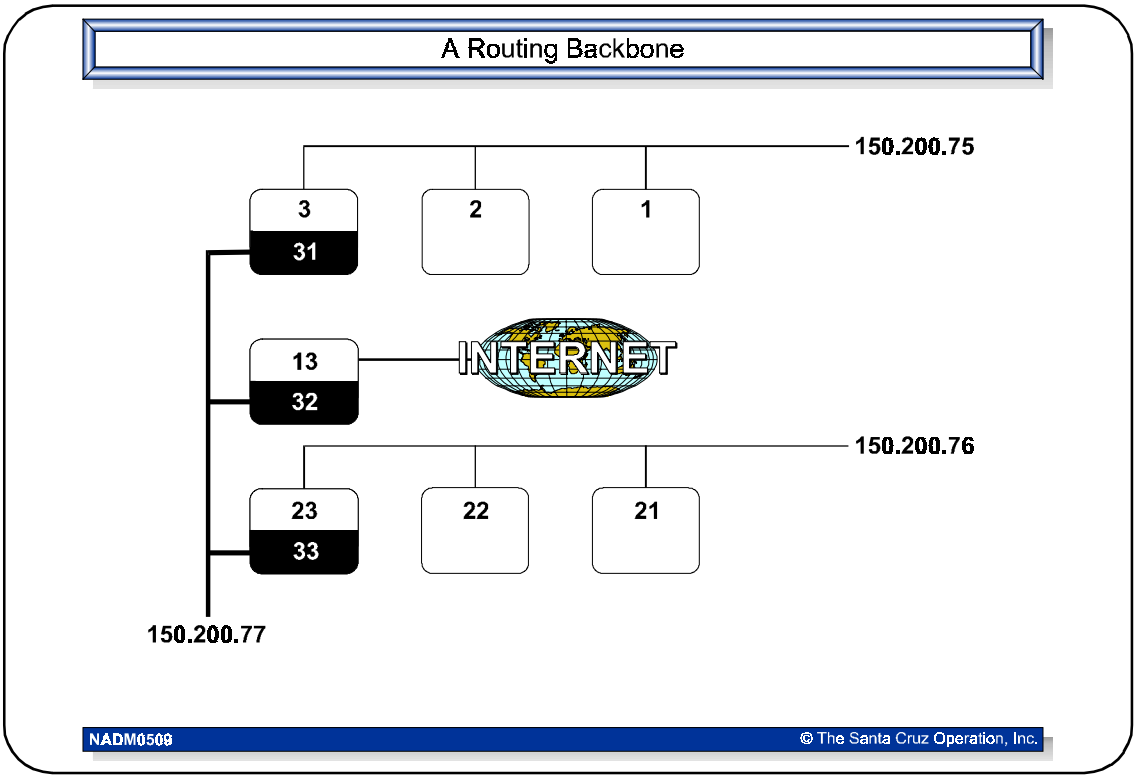
Automatically	When running the **Network Configuration Manager** to configure or remove a network interface. Through this process, a system always knows about the networks to which it is directly connected.
Manually	Using the **route**(ADMN) command, a system can be configured to use specified routes for communicating with networks to which it is not directly connected.

Using **route**(ADMN) for this purpose is feasible for small, static networks. It also uses fewer computer resources than using dynamic routing daemons.

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.

Routing daemons can reduce routing administration work on large or changing networks.

AV 5-9


A Routing Backbone
150.200.75
3
31
2
1
13
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0509
© The Santa Cruz Operation, Inc.

5.2.5 Manual updates and route(ADMN)

The **route**(ADMN) command is used to make 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

AV 5-9: A Routing Backbone
Use this slide to introduce the concept of a backbone as well as how the **route** command can be used to configure routes for hosts on such a network.

Given the network topology illustrated in AV 5-9, you can 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
  ```

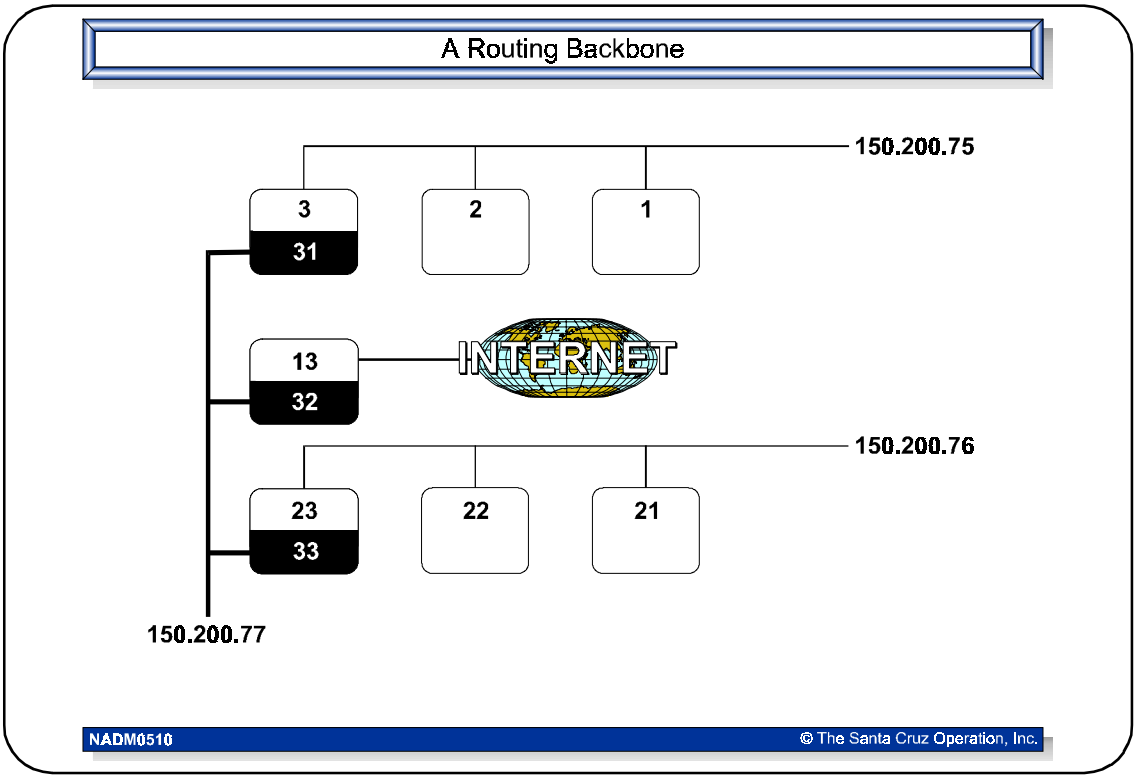
- Add a route to the 150.200.76 network for hosts 1 or 2 through host 3. Enter:

  ```
  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 5-25).

AV 5-10


A Routing Backbone
150.200.75
3
31
2
1
13
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0510
© The Santa Cruz Operation, Inc.

5.2.6 Default routes

AV 5-10: A Routing Backbone
Use this slide to illustrate when default routes are desirable.

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 they are directly attached to, and a default route
- Are very common in the LAN environment

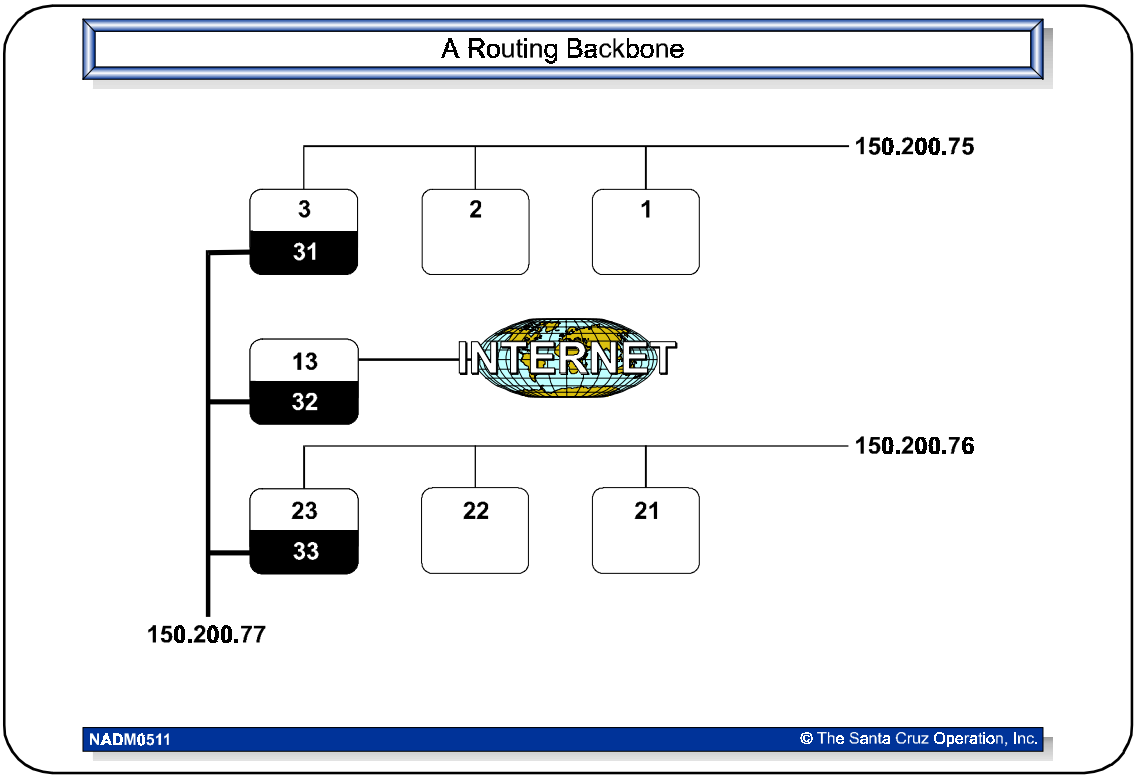
For example, consider AV 5-10:

- 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


A Routing Backbone
150.200.75
3
31
2
1
13
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0511
© The Santa Cruz Operation, Inc.

AV 5-11: A Routing Backbone
Use this slide to illustrate how default routes are created for host1 and host2.

Default routes are configured:

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

  ```
  route add default 150.200.75.3
  ```

- Permanently using an entry in the */etc/tcp* file that invokes **route**(ADMN). 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/tcp* file, place the following entry in the file:

  ```
  /etc/route add default 150.200.75.3
  ```

  See Section 7.1.4, “Multiuser-mode /etc/tcp activities,” on page 7-15 for more information on the */etc/tcp* file.

- Using the **Internet Manager** to add an entry in */usr/internet/etc/sco_ip/routes*. For example:

  ```
  net default 150.200.75.3
  ```

It is important that you do not set up more than one default route. Doing so can cause your network to hang.

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

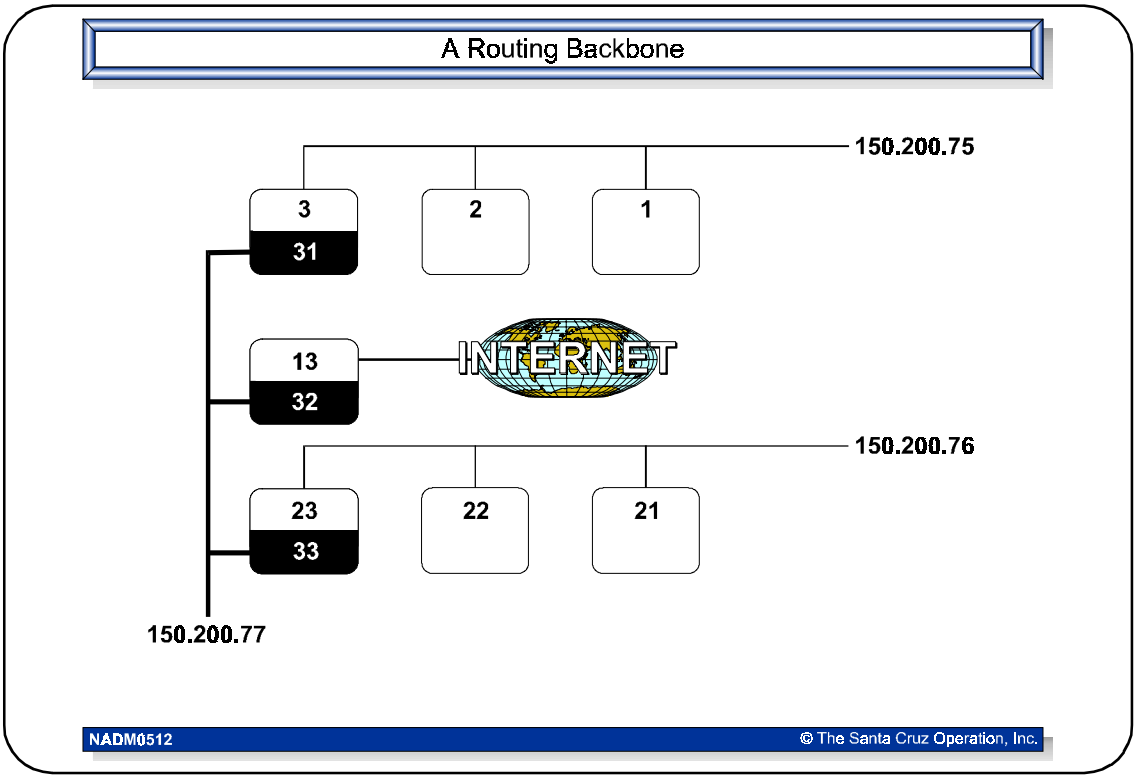
If your network topology changes regularly, or you would like your hosts to determine their routes automatically, you must configure your hosts to run a *routing daemon*. Note that using a routing daemon instead of static or default routes increases network traffic and CPU usage.

Routing daemons update routing tables dynamically. These routing daemons are provided with SCO OpenServer Release 5:

routed(ADMN)	Implements the Routing Information Protocol (RIP 1)
gated(ADMN)	Handles multiple routing protocols, including RIP, BGP, EGP and OSPF
irdd(ADMN)	Implements the Internet Router Discovery Protocol

Each of these daemons is discussed in the following sections.

AV 5-12


A Routing Backbone
150.200.75
3
31
2
1
13
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0512
© The Santa Cruz Operation, Inc.

5.3.1 routed(ADMN)

AV 5-12: A Routing Backbone
Use this slide to discuss the features and use of **routed**.

The **routed**(ADMN) daemon:

- Is started by default from an entry in the */etc/tcp* file, but only if **gated**(ADMN) cannot start. For further information, see Section 7.1.4, “Multiuser-mode /etc/tcp activities,” on page 7-15.

- Has several command-line options that control its behavior. Refer to **routed**(ADMN) for additional information.

When **routed** starts, it:

1. Broadcasts on each local network to advertise its routes to any other nodes directly attached to its networks.

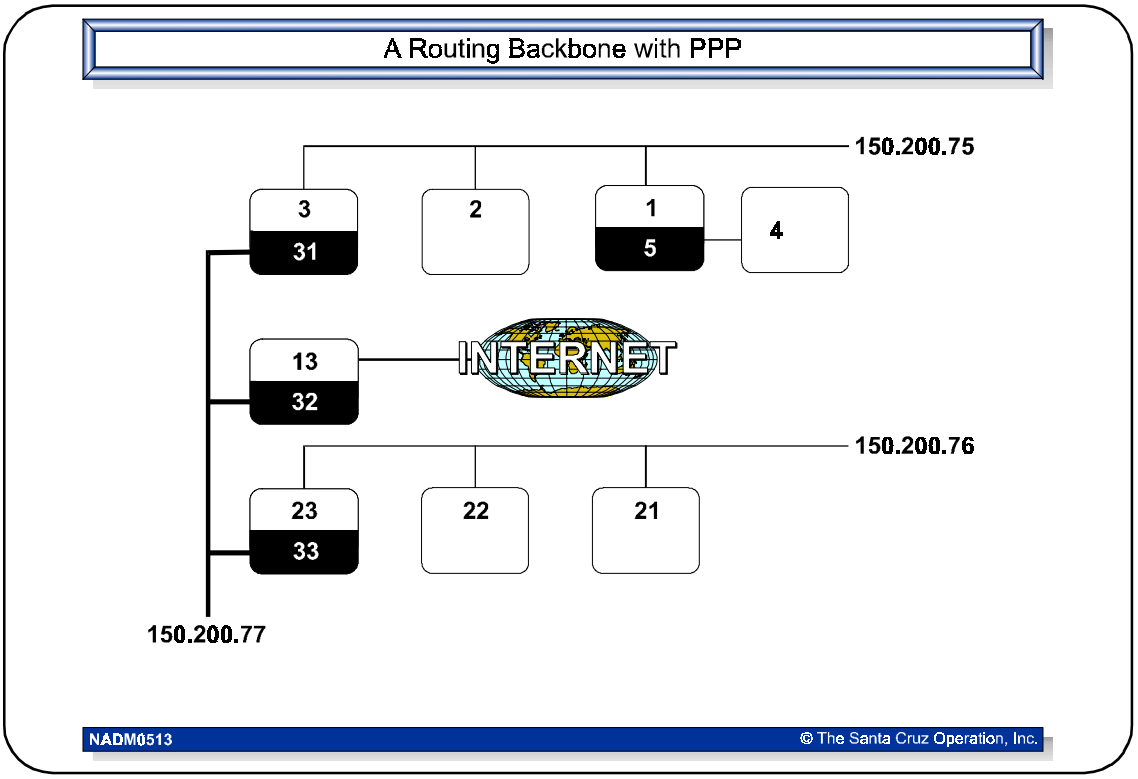
2. If other routers exist they continually cooperate in maintaining a globally consistent view of routing by exchanging routing data using one of the routing protocols mentioned above.

For example, in AV 5-12, routing daemons on hosts 3/31, 13/32, and 23/33 can all exchange routing information.

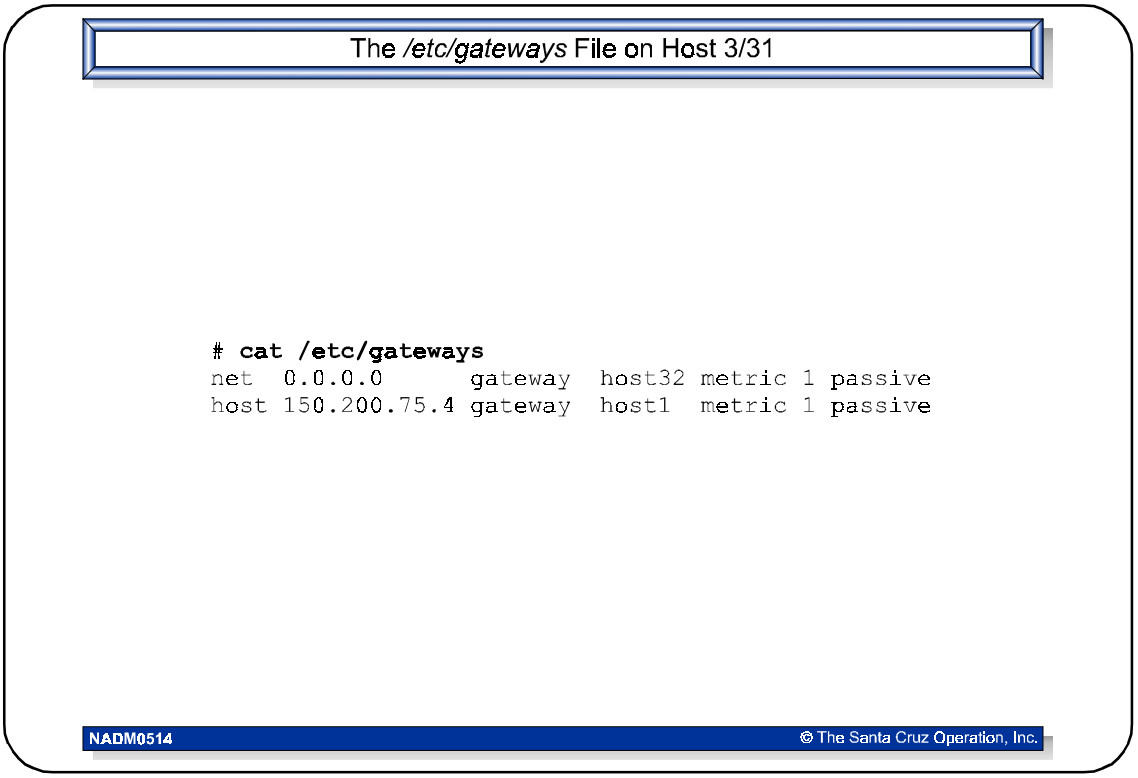
Remember, hosts 1, 2, 21, and 22 do not need to run **routed**. They can use a default route to hosts 3 and 23 respectively.

If hosts 31 and 33 list host 32 as a default router, host 32 does not need to tell hosts 31 and 33 about its connection to the Internet. However, host 32 still needs to run a routing daemon so that it can learn the routes to the 150.200.75 and 150.200.76 networks from hosts 31 and 33 respectively.

AV 5-13


A Routing Backbone with PPP
150.200.75
3
31
2
1
5
4
13
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0513
© The Santa Cruz Operation, Inc.

AV 5-14


The /etc/gateways File on Host 3/31
cat /etc/gateways
net 0.0.0.0   gateway host32 metric 1 passive
host 150.200.75.4 gateway host1 metric 1 passive
NADM0514
© The Santa Cruz Operation, Inc.

AV 5-13: A Routing Backbone with PPP
Use this slide to illustrate entries in the */etc/gateways* file in the next slide.
NOTE: This slide has one difference from the previous routing backbone slides. Now, there is a *host4* connected to *host1/5* on the 150.200.75 network. This kind of attachment is possible with either PPP or SLIP. Students are more likely to have heard of PPP than SLIP, so the connection is assumed to be a PPP connection.

In some versions of SCO OpenServer Release 5, **routed** does not read the */etc/gateways* file and update the routing table.

Normally, the TCP/IP startup procedure and the subsequent protocol exchanges by routers will find and keep routes up to date. You only need to create the */etc/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 hosts 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 and suppose that:

  - The link between host 4 and1/5 is through a point-to-point protocol, such as PPP or SLIP.

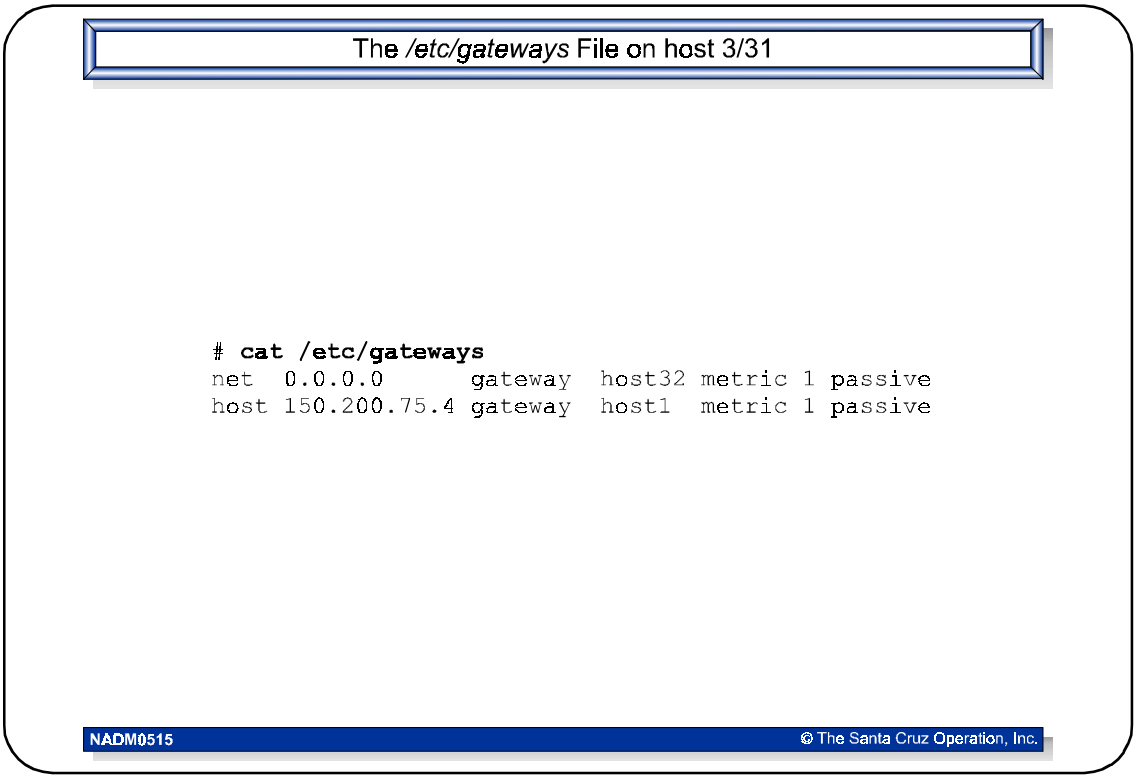
  - Host 1/5 is not running a routing daemon.

  The route to host 4 is not known to host 3/31.

AV 5-14: The */etc/gateways* File on host 3/31
This slide shows a sample entry in the */etc/gateways* file on host 3/31. The host route causes the route to host 4 to be visible to host 3/31 even if host 1/5 is not running a routing daemon or performing proxy ARP.

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

AV 5-15


The /etc/gateways File on host 3/31
cat /etc/gateways
net 0.0.0.0   gateway host32 metric 1 passive
host 150.200.75.4 gateway host1 metric 1 passive
NADM0515
© The Santa Cruz Operation, Inc.

AV 5-15: The */etc/gateways* File on host 3/31
This slide repeats AV 5-14, showing a sample entry in the */etc/gateways* file on host 3/31. The host route causes the route to host 4 to be visible to host 3/31 even if host 1/5 is not running a routing daemon or performing proxy ARP.

Each entry in the */etc/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.
`name1`	Is the symbolic name of the destination network or host (from */etc/hosts*), 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 *val* field essentially ranks multiple routes to the same destination. The software chooses the best route.

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 through RIP exchanges.
`external`	The gateway does not communicate directly. The route is not entered into the kernel's routing table until a RIP or IRDP packet proposes the route. This is useful in situations where you do not know the router protocol used on the gateway and do not expect routing information from it.

5.3.2 gated(ADMN)

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

- You are using your SCO OpenServer Release 5 system as a router to an autonomous system (AS).

If students don't know what an AS is, they probably are not responsible for one. Be prepared to explain what an AS is if students persist in wanting to know what one is.

- There are more than 15 hops in the route to a destination network.
- 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**(ADMN) 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.

NSF-net is the portion of the Internet funded by the United States' National Science Foundation (NSF).

- Open Shortest Path First (OSPF). This protocol is best suited for large, complex intranets with many routers. It includes features such as:

  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 using the 100Mbps connection and other data might be routed using a 10Mbps connection.

  Scalability
  : For the above and other reasons, OSPF scales better than the protocols supported by **routed**.

You can also configure **gated** to use RIP, just as **routed**(ADMN) does.

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

5.3.3 irdd(ADMN)

The **irdd**(ADMN) daemon is an implementation of the Internet Route Discovery Protocol. It is best used in small networks within autonomous sytems. It provides a low-overhead method for non-routers to add default routes.

On systems configured as routers, **irdd** advertises the system periodically so that any hosts on the network running router discovery are aware of the router. It also answers any solicitation requests received from newly initialized hosts.

The behavior of **irdd** can be tuned using its configuration file, */etc/irdd.conf*. Refer to **irdd**(ADMN) and *irdd.conf*(SFF) for more information.

The */etc/tcp* startup script normally starts and stops **irdd** by default if */etc/irdd* is executable and */etc/irdd.conf* exists.

5.4 Troubleshooting Routes

In addition to the **netstat**(ADMN) command (discussed in Section 5.2.3, “The netstat(ADMN) command,” on page 5-19), there are several commands and files that you can use to troubleshoot routing related problems. These include:

- **ping**(ADMN)
- **traceroute**(ADMN)
- **ripquery**(ADMN)

5.4.1 The ping(ADMN) command

Recall from Lesson 3: “Configuring a Network Adapter,” that you can use the **ping**(ADMN) command to verify connectivity of the hosts on your network. (See Section 3.2.2, “The ping(ADMN) command,” on page 3-16.)

In addition to verifying that routes are working between two hosts, you can use the **-R** option of **ping** to record and display routes between hosts. For example, this command line sends a single packet from *firefly* to *mantis* and displays the route:

```
ping -R -c1 150.200.75.1
PING 150.200.75.1 (150.200.75.1): 56 data bytes
64 bytes from mantis (150.200.75.1): icmp_seq=0 ttl=64 time=0 ms
RR:   spider1 (150.200.75.3)
      mantis (150.200.75.1)
      spider2 (150.200.76.7)
      firefly (150.200.76.12)

--- 150.200.75.1 ping statistics --- 1 packets transmitted, 1 packets
received, 0% packet loss round-trip min/avg/max = 0/0/0 ms
```

If the **ping** does not work, follow these steps:

(Note the first three steps were discussed in Section 3.2, “Testing TCP/IP Connectivity,” on page 3-15).

Step 1. Verify the local protocol stack is working. Ping *localhost*.

Step 2. Verify the network adapter is working. Ping the broadcast address of the network adapter.

Step 3. Verify the router is receiving our packets. Ping the IP address of the router that shares the same network address as the host you are pinging from.

Step 4. Verify the router is forwarding packets. Ping the IP address of the router that does not have the same network address as the host you are pinging from. 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 5-11)

If you have not yet isolated the problem, use the **traceroute**(ADMN) command.

5.4.2 The traceroute(ADMN) command

Use the **traceroute**(ADMN) 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:

`traceroute [-`*`options`*`] `*`host`*

For example:

```
traceroute ftp.sco.com
1  ts018e01.hil-ny.concentric.net (206.173.18.5)  240 ms  230 ms  230 ms
2  rt001e0101.hil-ny.concentric.net (206.173.18.1)  240 ms  240 ms  230 ms
3  206.83.91.173 (206.83.91.173)  330 ms  320 ms  320 ms
4  us-hq-core1-f1-0.rtr.concentric.net (205.158.0.4)  330 ms  330 ms  320
ms
5  us-ca-pa-core1-a1-0d5.rtr.concentric.net (207.88.0.14)  320 ms  320 ms
320 ms
6  paix.exodus.net (198.32.176.15)  320 ms  320 ms  320 ma
7  scca-01-h0-0-0.core.exodus.net (209.1.169.229)  320 ms  320 ms  310 ms
8  core3-f0-1-0.exodus.net (209.1.169.14)  330 ms scca-01-h0-0-
0.core.exodus.net (209.1.169.229)  330 ms  320 ms
9  sco.com (209.1.239.198)  330 ms  330 ms  320 ms
10  virtualweb.sco.com (209.1.8.14)  330 ms  330 ms sco.com
(209.1.239.198)  330 ms
```

Refer to the **traceroute**(ADMN) manual pages for more information.

5.4.3 The ripquery(ADMN) command

The **ripquery**(ADMN) 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.

For example, to send a version 1 RIP POLL packet from *mantis* to request all the routes from the gateway *spider1*, enter:

```
ripquery -1 -p spider1
64 bytes from spider1(150.200.75.3) to 150.200.75.1 version
1:
     206.173.18.5     metric  1
     150.200.76.0     metric  1
     150.200.75.0     metric  1
```

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 are the steps a system takes to route an IP packet to a system not directly connected to the local network segment?

3. What is the kernel routing table? How does it get updated?

4. What are the routing daemons supported by SCO OpenServer Release 5?

5. 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.

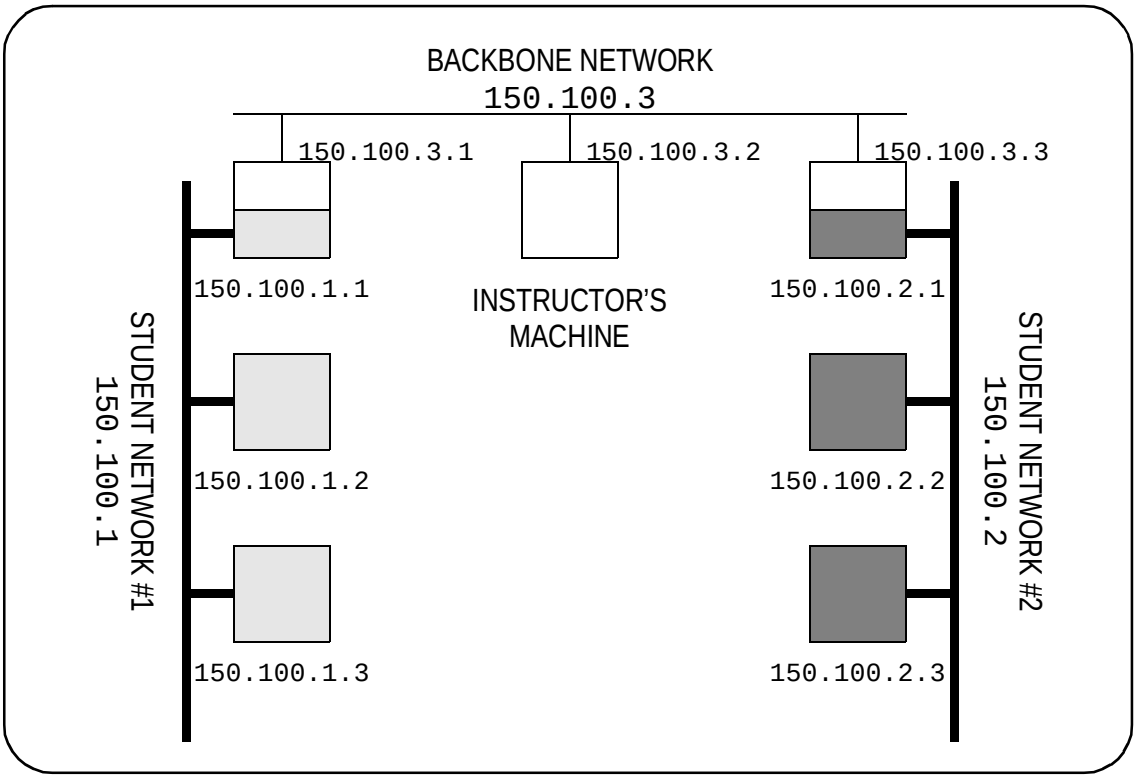
5.6.1 Basic router setup

Your instructor will provide you with the information needed 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 has if it is going to become a router.

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

Draw a diagram on the board which shows the network topology of the classroom. This diagram might resemble the following topology:



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 3-11. 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 5-9).

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 3-24.

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 the */etc/hosts* file 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 using **vi**(C) to edit the */etc/hosts* file. Refer to Section 2.4.3, "The /etc/hosts file," on page 2-33 for details.

Step 4. Test connectivity using the tools described in Section 5.4, "Troubleshooting Routes," on page 5-38. (HINT: Do not forget to use **netstat** with the **-r** option).

Remember that the router must be configured correctly and running in multiuser mode before you are able to reach systems on other networks.

5.6.2 The `ipforwarding` parameter

The objective of this lab is to observe the effects of the `ipforwarding` kernel parameter. You can 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 **Network Configuration Manager**.

Step 1. Use **inconfig**(ADMN) to verify the value of `ipforwarding` is set to 1 on the router systems:

```
inconfig -v ipforwarding
```

Step 2. Observe that packets flow through the router. Use **ping** from a host on one side of the router to contact 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. Use the **ping**(ADMN) and **rlogin**(TC) commands.

On non-routers, can you connect to systems in other networks? Why or why not?

On the routers, can you connect to systems in other networks? Why or why not?

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

```
inconfig -v ipforwarding 1
```

Step 6. Observe that packets flow through the router. Use ping from a host on one side of the router to contact 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(ADMN)	Is a routing daemon that handles multiple routing protocols
inconfig(ADMN)	Is a configuration utility for the default kernel parameters used by the TCP/IP protocol suite
netconfig(ADM)	Configures networking products and can be invoked as the **Network Configuration Manager**
netstat(ADMN)	Examines currently configured network interfaces and other network-related data structures, including routing table information
ping(ADMN)	Verifies connectivity of the hosts on a network
ripquery(ADMN)	Requests all routes known by a RIP gateway
route(ADMN)	Manually manipulates the routing tables
routed(ADMN)	Is a routing daemon that implements the Routing Information Protocol (RIP)
traceroute(ADMN)	Traces and displays the route an IP packet follows to another Internet host

Relevant files

/etc/default/inet	Configuration file for the **inconfig**(ADMN) command documented in the *inet*(SFF) manual page.
/etc/inetd.conf	Configuration file for TCP/IP daemons documented in the *inetd.conf*(SFF) manual page.

5.8 Answers to Review Questions

This section provides suggested 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. 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 finds that the ultimate destination can be reached through the local gateway.

3. 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**(ADMN) command; and dynamically, using specialized routing daemons running on the system.

4. The routing daemons supported by SCO OpenServer Release 5 are:

   - **irdd**(ADMN)—implements the Internet Router Discovery Protocol
   - **routed**(ADMN)—implements version 1 of the Routing Information Protocol (RIP I)
   - **gated**(ADMN)—handles the RIP (I or II), EGP, BGP, and OSPF routing protocols

5. You can use the **netstat**(ADMN), **traceroute**(ADMN), **ping**(ADMN), and **ripquery**(ADMN) utilities to monitor routing on your system.

6

Configuring WAN Connections

Instructor Guide

Lesson 6

Configuring WAN Connections

Suggested lesson duration

This lesson should take approximately 6 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

This lesson requires:

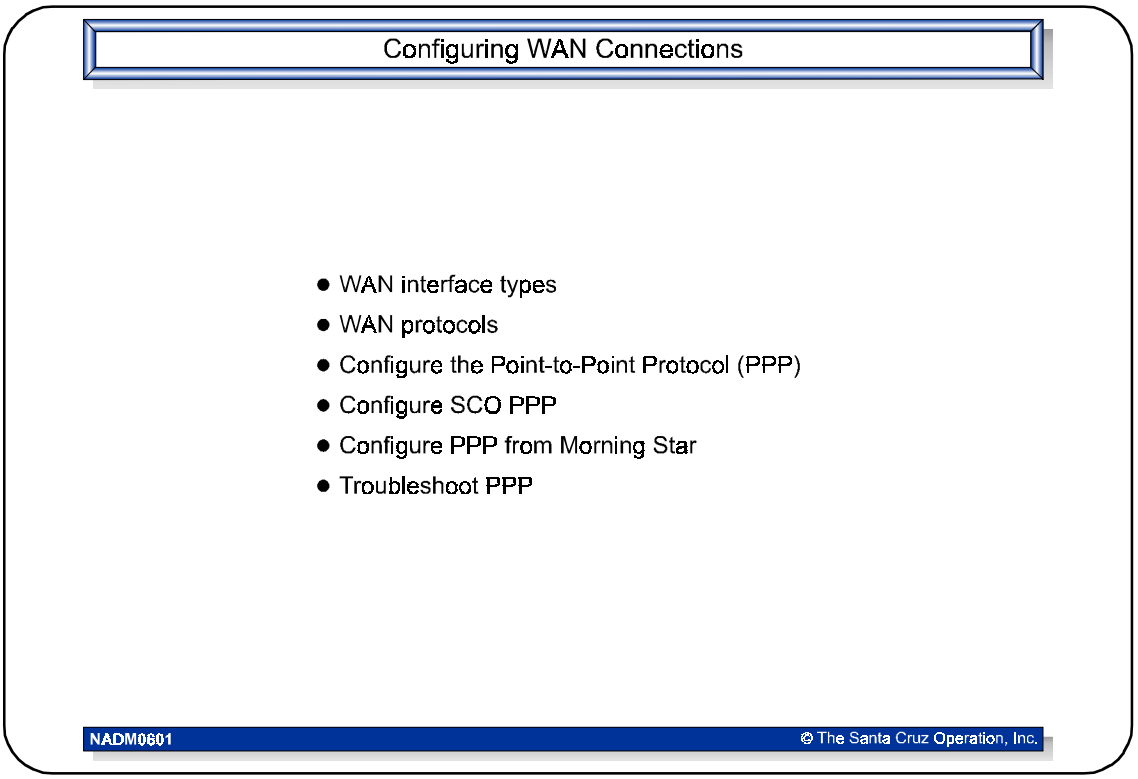
- Modems
- Two(2) or more phone lines for PPP
- Copies of the Example PPP Topology slide for students to use as a reference throughout the lesson. See AV 6-9 on page 6-22.

Instructor preparation

Have telephone rates for normal calls, leased lines (T1 and T3), ISDN, cable modems, ADSL, and frame relay available for discussions during the discussion of those technologies.

Familiarize yourself with setting up UUCP devices and PPP connections. A thorough discussion of the topics related to PPP can be found in **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Point-to-Point Protocol (PPP).**

AV 6-1


Configuring WAN Connections
WAN interface types
WAN protocols
Configure the Point-to-Point Protocol (PPP)
Configure SCO PPP
Configure PPP from Morning Star
Troubleshoot PPP
NADM0601
© The Santa Cruz Operation, Inc.

Introduction

Until now, this course has focused on local area networks using Ethernet cards. Your site might have alternate requirements, such as:

- A need to connect to the Internet
- Multiple geographically distinct LANs that need to be connected to form a wide area network (WAN)
- Telecommuters

This lesson discusses different networking interfaces and focuses on the use of PPP to connect LANs to:

- Other LANs
- Hosts
- The Internet

Objectives

This lesson discusses different LAN and WAN interfaces. By the end of this lesson, you will be able to:

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

AV 6-1: Configuring WAN connections
This first slide describes the objectives covered in this lesson.

6.1 WAN Interface Types

Wide Area Networks (WANs) connect hosts 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 because multiple bits can be encoded into each signal.
- Often used incorrectly as a synonym for BPS. BPS is a measure of transmission speed.

Older modulation standards encoded one bit per signal making the baud rate and bps equal. This leads to the terms being used interchangeably. To achieve rates higher than the 3000 Hz bandwidth that typical phone lines have, newer modulation standards encode multiple bits into each signal achieving higher bps.

AV 6-2

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.

NADM0602 © The Santa Cruz Operation, Inc.

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.

Mention that there might be additional overhead due to the protocols lowering the effective data transmission rate. For example, PPP uses headers which increase the overhead and reduce data throughput.

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.

Clarify the difference between latency and speed. Explain how latency can also affect performance. A good analogy for explaining latency is an international phone call where there is a noticeable delay in response. You can still talk as fast but the delays waiting for responses usually makes the entire conversation take longer.

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:

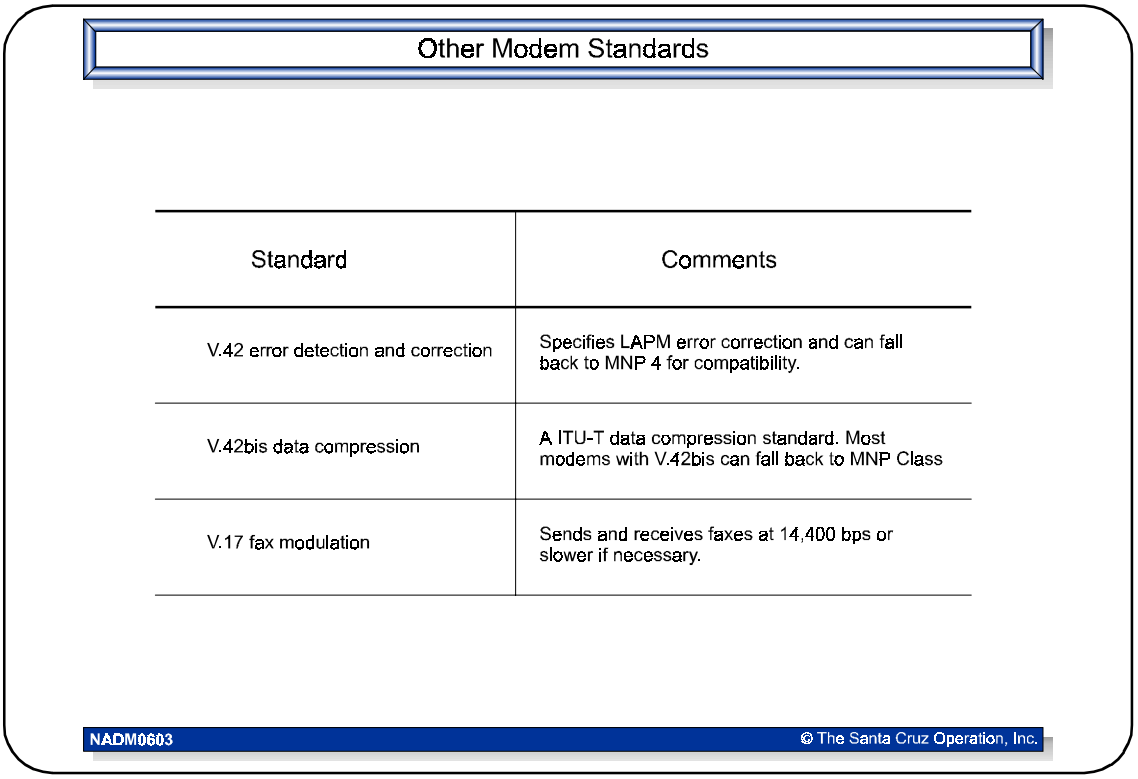
The ITU-T was formerly the International Telegraph and Telephone Consultative Committee (CCITT).

- 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 6-2: Modem Modulation Standards
This slide shows the modem standards for modulation and the maximum speed modems can achieve using these standards.

The suffix “bis” on a standard means “the second”. Similarly “ter” means “the third.”

AV 6-3



Instructor Guide

- 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.

AV 6-3: Other Modem Standards
Display modems standards for error correction, data compression, and fax transmission.

Link Access Protocol for Modems (LAPM).

Microcom Networking Protocol (MNP)—A standard for compression, error detection and correction. This is a proprietary standard licensed by most modem manufacturers.

V.XXX is the series of ITU-T standards for telecommunications.

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 talk to each other.

- The speed the computer talks to the modem can also be configured.

  Configure this to be higher than the modem's maximum connection speed to accommodate any compression the modem is using.

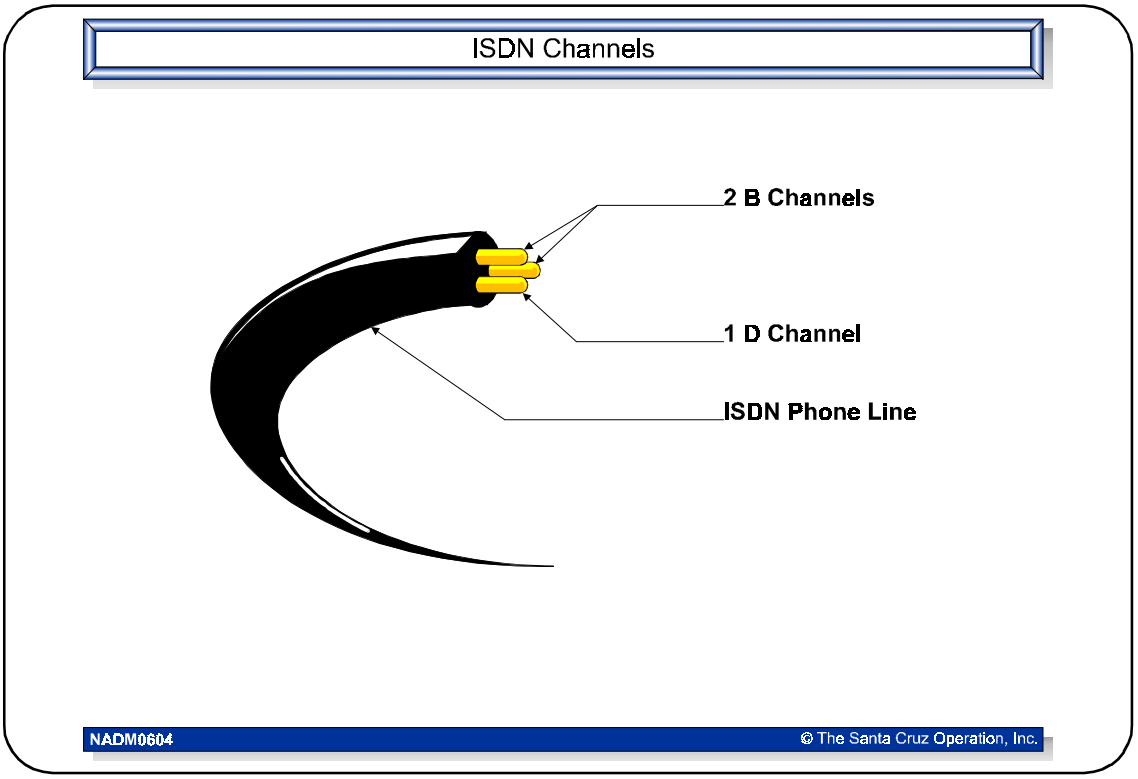
When contact is initiated between modems, they:

1. Negotiate which standards they can both support.

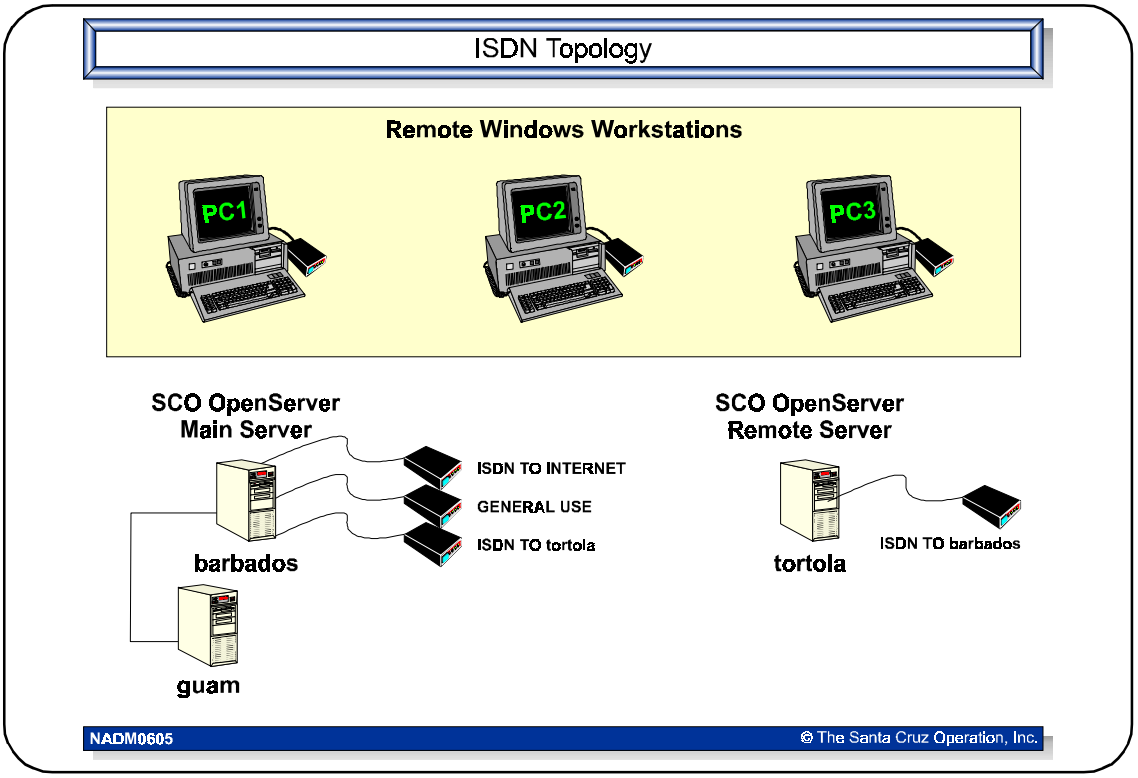
2. Determine the speed the line conditions can support.

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


ISDN Channels
2 B Channels
1 D Channel
ISDN Phone Line
NADM0604
© The Santa Cruz Operation, Inc.

AV 6-5


ISDN Topology
Remote Windows Workstations
PC1
PC2
PC3
SCO OpenServer
Main Server
ISDN TO INTERNET
GENERAL USE
ISDN TO tortola
barbados
guam
SCO OpenServer
Remote Server
ISDN TO barbados
tortola
NADM0605
© The Santa Cruz Operation, Inc.

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.

In the US, voice lines are sampled 8K times per second. Each sample contains 8 bits. Thus, normal voice phone lines operate at 64Kbps, as does each ISDN channel.

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

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

AV 6-4: ISDN Channels
Use this slide to illustrate how ISDN channels can work together.

Three common ISDN devices in use include:

- 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.

A common mistake is to refer to these devices as ISDN modems. They do not modulate or demodulate signals but instead transmit information digitally. This term, though inappropriate, has gained acceptance because these devices serve a similar function to modems—sending digital information over phone lines.

- 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 to 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*.

AV 6-5: ISDN Topology
Versions of this slide are used throughout this lesson as a reference for the various methods for connecting the machines together. In this scenario:

- There is a main office that has a LAN with the machines *barbados* and *guam.*
- *barbados*, the main company server, is:
  - Used as the gateway to the Internet
  - Connects to *tortola*, which is in a remote office, using a dedicated ISDN adapter
  - Allows dialin access from the others

There are three PCs (PC1-3) that are used by telecommuters or people who travel. How these PCs contact barbados are discussed in Section 6.2.2, “Point-to-Point Protocol (PPP),” on page 6-23.

Spend some time going over this slide with the students. Use it to illustrate the use of ISDN between the ISP, *tortola*, and *barbados*, with standard modems in use for the *barbados* and the PCs. The PCs might be connected using UUCP or some other interface technique, such as PPP.

6.1.3 Other digital subscriber line technologies

Several digital WAN technologies not directly supported by SCO OpenServer, include:

- 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 email.
- VDSL—Very high data rate Digital Subscriber Line is similar to ADSL but provides faster data rates at shorter cable lengths.
- HDSL—High date 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 muliplexed together	64Kbs * the number of lines
Fractional T2	FT3	Any integer multiple, up to 27, of 1.5Mbps T1 lines muliplexed together	1.5Mbs * the number of lines

NADM0606 © 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 muliplexed together	64Kbs * the number of lines
Fractional E2	Any integer multiple, up to 3, of E1 lines muliplexed together	2.048Mbs * the number of lines

NADM0607 © The Santa Cruz Operation, Inc.

6.1.4 Leased lines

Leased telephone lines:

- Are point to point connections, like ISDN
- Require special telephone equipment called a CSU/DSU that 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.

Explain how the fractional rates work and the difference between them and the *official* rates.

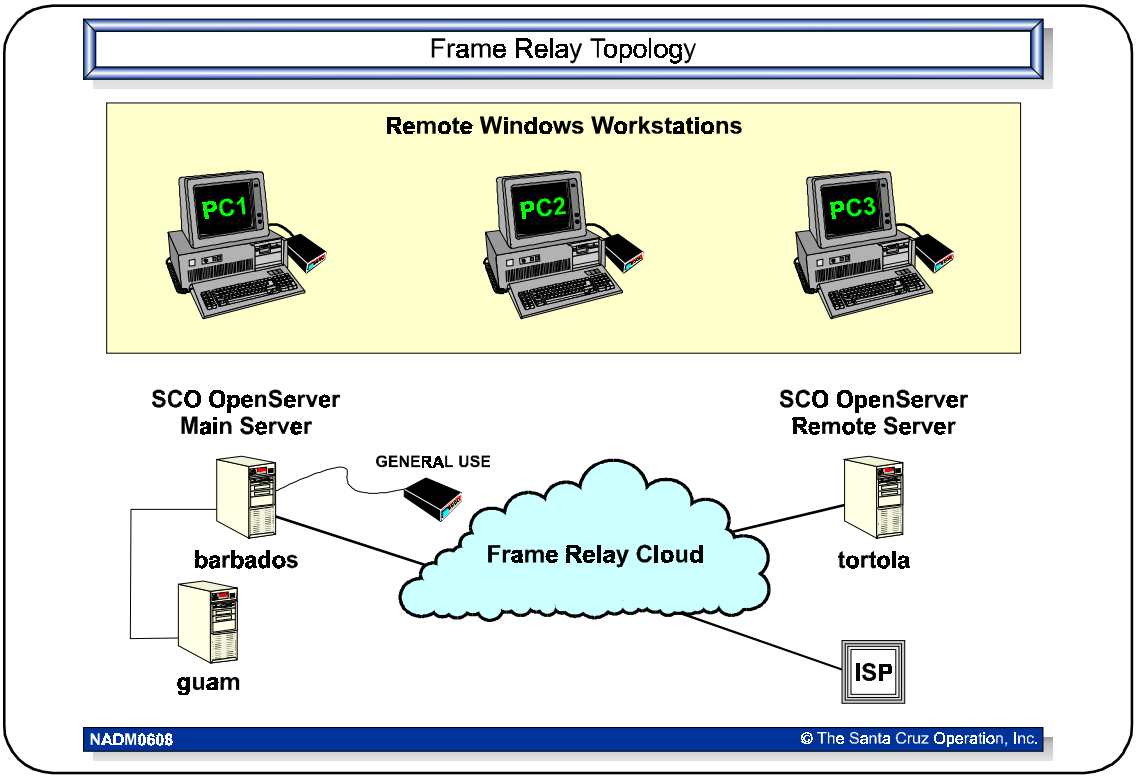
AV 6-6: Leased Line Speeds in North America and Japan
This slides shows the leased line speeds available in North America and Japan.

AV 6-7: Leased Line Speeds Elsewhere in the World
This slides shows the leased line speeds available in areas outside North America and Japan.

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.

Discuss leased line rates and availability in your area.

AV 6-8


Frame Relay Topology
Remote Windows Workstations
PC1
PC2
PC3
SCO OpenServer
Main Server
GENERAL USE
SCO OpenServer
Remote Server
barbados
Frame Relay Cloud
tortola
guam
ISP
NADM0608
© The Santa Cruz Operation, Inc.

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.

AV 6-8: Frame Relay Topology
Use this slide to demonstrate how frame relay can be used to connect all of the machines together with a single frame relay connection for each location. This scenario is really only better than ISDN if:

- One or more of the PCs and/or *tortola* are continually connected.
- A speed is needed that is faster than what ISDN can provide.
- You have multiple satellite offices to connect to a central office.
- You want to save long distance charges on connections separated by large distances.

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 through 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.
- It requires special hardware from the phone company. This hardware 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. 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.

For full-time connections, frame relays can be a better choice because they are charged flat rates. Discuss frame relay and ISDN rates and availability in your area.

SCO OpenServer currently does not directly support frame relay. Frame relay devices connected to a LAN can still be used.

6.1.6 Cable modems

Cable modems:

- Use existing coax wiring. This is currently being used for cable television to also carry data communications
- Allow regular television programming and digital data to be transmitted simultaneously
- Currently have no industry wide standards so performance and operation of cable modems vary between different implementations.

Instructor Guide

Bandwidth can vary considerably because each cable company must decide which channels and how many channels to devote to data transmission.

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.

Point out how this could pose a security risk through packet sniffing because all users on a segment can see all the packets broadcast on that segment, not just their own. Packet sniffing is discussed in Section 10.4.1, “Packet sniffing,” on page 10-27.

SCO OpenServer currently does not support cable modems. Cable modems connected to a LAN can still be used.

6.2 WAN Protocols

The physical differences between WAN and LAN hardware requires that WAN connections use separate protocols to handle the unique features of 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.

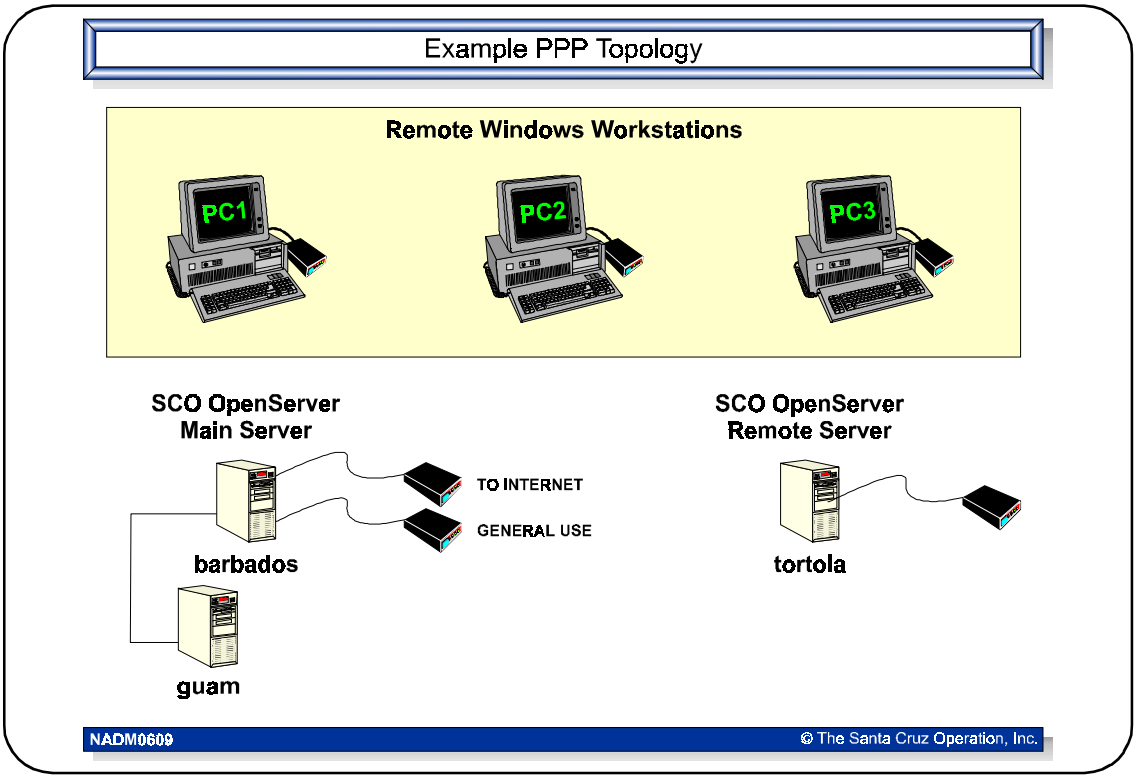
Modern SLIP implementations can have many PPP features, including:

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

Note: The SCO OpenServer 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


Example PPP Topology
Remote Windows Workstations
PC1
PC2
PC3
SCO OpenServer
Main Server
TO INTERNET
GENERAL USE
barbados
guam
SCO OpenServer
Remote Server
tortola
NADM0609
© The Santa Cruz Operation, Inc.

6.2.2 Point-to-Point Protocol (PPP)

The most popular protocol to use over WAN interfaces is the Point-to-Point Protocol (PPP).

PPP provides TCP/IP services over WAN connections:

While many WAN devices (e.g., ASCEND or CISCO routers) support IPX and other protocols over PPP, SCO OpenServer Release 5 only supports IP over PPP.

- Allowing the transmission of packets of any network protocol (for example, IP, IPX, IPv6) 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 how to answer the modem for incoming connection requests
- Monitoring the line and dropping the connection if no packets are sent for some period of time

Lead a discussion with students about their experiences with PPP. Also, indicate how your company is connected to the Internet: PPP, ISDN, and so on.

AV 6-9: Example PPP Topology
This slide is a modification of the previous ISDN and Frame Relay Topology slides and is used as a reference for the various kinds of things that can be done with PPP. The scenario is the same as before, except that only PPP connections are made through the modems shown in the slide.

Spend some time going over this slide with the students. Use this example as the foundation for a discussion of how your students might use PPP. Make sure your discussion goes beyond the (implied) dynamic connections shown here to include manual outgoing and dedicated PPP lines. These terms (dynamic, manual, and dedicated) are used throughout the lesson as well, so this is a good time to introduce them to students.

Hand out the copies of AV 6-9. You should keep this slide handy because it is referred to throughout most of this section.

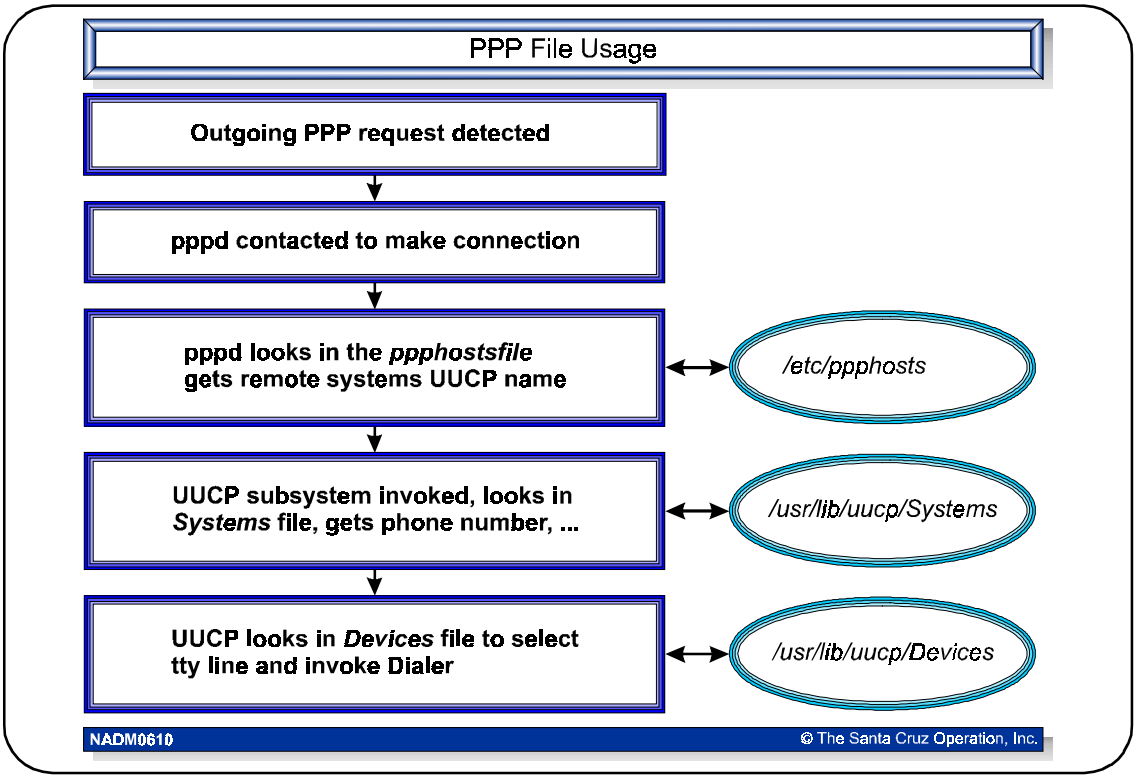
SCO OpenServer Release 5 includes two versions of PPP. Both support the transmission of IP packets using TCP/IP applications such as **rlogin**(TC) and **telnet**(TC) over serial lines.

SCO PPP	Installed by default and configured using the **Network Configuration Manager**

PPP from Morning Star	Separately installed from the OpenServer CD-ROM and configured using the **Internet Manager** PPP from Morning Star provides enhanced security features including: • Link level peer authentication and periodic re-authentication • Advanced packet filtering • Virtual Private Networks

PPP from Morning Star also allows the remote server to assign IP addresses with both manual and dynamic outgoing connections. For versions of SCO OpenServer before 5.0.4, SCO PPP only supported this *dynamic IP address* feature with manual outgoing connections.

AV 6-10


PPP File Usage
Outgoing PPP request detected
pppd contacted to make connection
pppd looks in the ppphostsfile gets remote systems UUCP name
/etc/ppphosts
UUCP subsystem invoked, looks in Systems file, gets phone number, ...
/usr/lib/uucp/Systems
UUCP looks in Devices file to select tty line and invoke Dialer
/usr/lib/uucp/Devices
NADM0610
© The Santa Cruz Operation, Inc.

6.3 PPP Configuration Overview

Both SCO PPP and PPP from Morning Star support incoming, outgoing, and dedicated connections. These are referred to as *link types*, or *endpoints:*

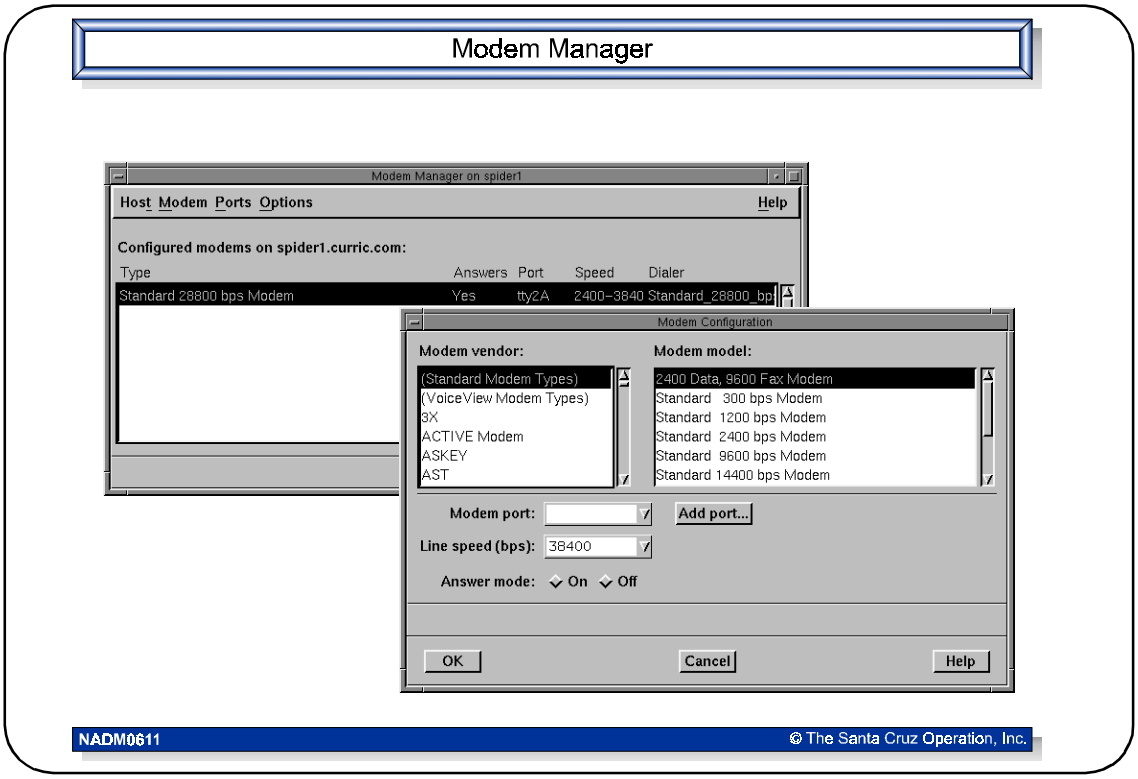
Incoming	Accepts PPP connection requests from remote hosts; sometimes referred to as *dynamic incoming*
Outgoing	Starts an outgoing connection initiated either: • Dynamically—whenever an outgoing IP packet is present (for the specified remote host) • Manually—using the **pppattach** command
Dedicated	Uses a dedicated serial line connecting a remote host; most commonly used with leased phone lines between locations

AV 6-10: PPP File Usage
This slide illustrates how PPP uses the PPP and UUCP configuration files.

While the specific procedures and file names differ depending on which version is installed, PPP configuration involves these basic steps:

Configure the hardware	Includes these tasks: • Configure the modem • Configure the serial port
Configure UUCP	Configure the UUCP *Systems*(F) file to add an entry for outgoing connections
Configure PPP	Includes these tasks: • Link the PPP driver into the kernel • Add the TCP/IP->PPP protocol chain • Modify the PPP configuration files *ppphosts*, *pppauth*, and *pppfilters*

AV 6-11



Instructor Guide

6.4 Configuring SCO PPP

Configuring a basic connection using SCO PPP requires these steps:

Step 1. Configure the modem and serial port using the **Modem Manager.**

Step 2. Add an entry to */usr/lib/uucp/Systems* (for outgoing connections).

Step 3. Configure PPP using the **Network Configuration Manager.**

All PPP configuration tasks are performed using **netconfig** in this guide. The **PPP Manager** can be used to add and modify connections after the PPP driver and TCP/IP protocol have been configured with **netconfig**. However, the interface is different from that used by **netconfig**. For consistency and simplicity, this guide uses the same interface for all PPP configuration tasks.

Also note that at the time of writing, none of the graphical tools properly configure */usr/lib/uucp/Systems*. Hence the manual configuration steps used in this guide.

While configuring advanced options of PPP can be a complex task, often PPP connections can be configured with a minimum set of basic parameters. The sections that follow describe simple PPP connections.

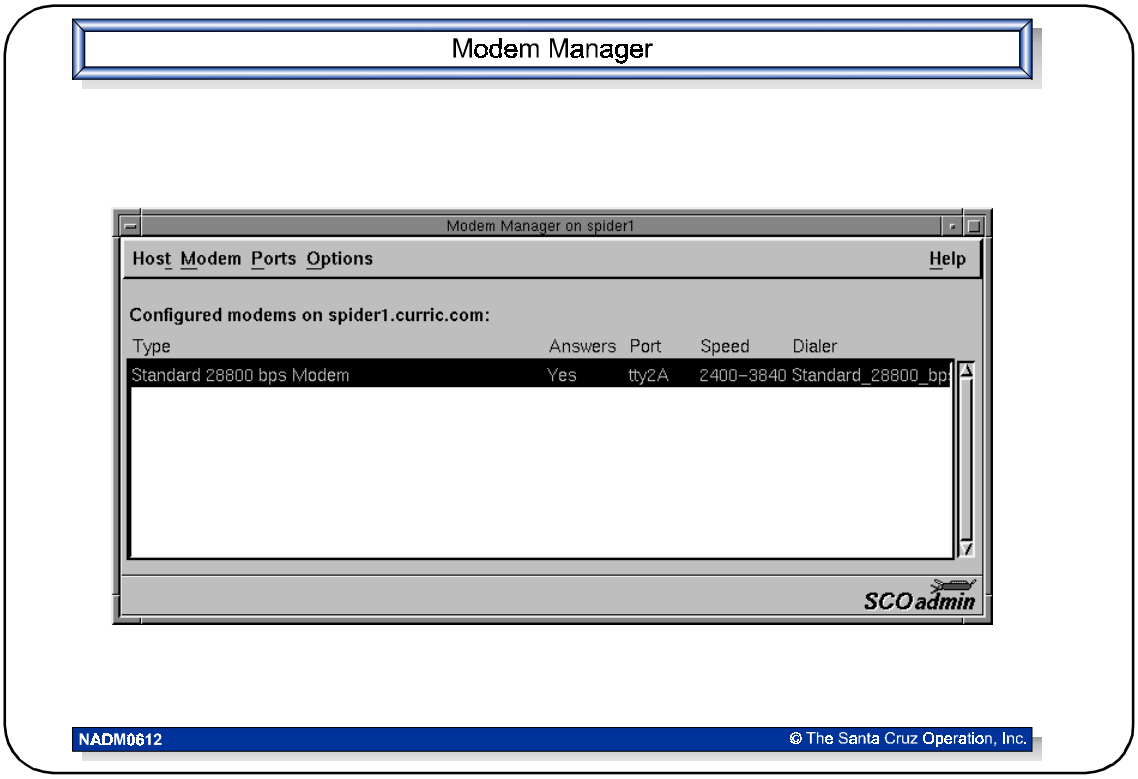
6.4.1 Configuring the modem

AV 6-11: The Modem Manager
This is a screen shot of the **Modem Manager** and the **Add Modem** window.

The **Modem Manager** configures your modem by performing these tasks:

- Configuring the serial port characteristics, such as speed
- Disabling or enabling the modem control port (*tty1A*, for example)
- Adding a dialer entry for the modem in */usr/lib/uucp/Devices*

AV 6-12



Instructor Guide

AV 6-12: The Modem Manager
This is a screen shot of the **Modem Manager**.

To configure a modem:

Step 1. Start the **Modem Manager** as **root**.

Step 2. Select **Automatic Detection** or **Manual Configuration**.

With automatic detection, the system queries the ports to find the make and model of your modems. Manual configuration requires you to specify the vendor, model, and port of your modem.

Step 3. Select which ports you want to scan and click on **Detect**.

If automatic detection fails, proceed to Step 4.

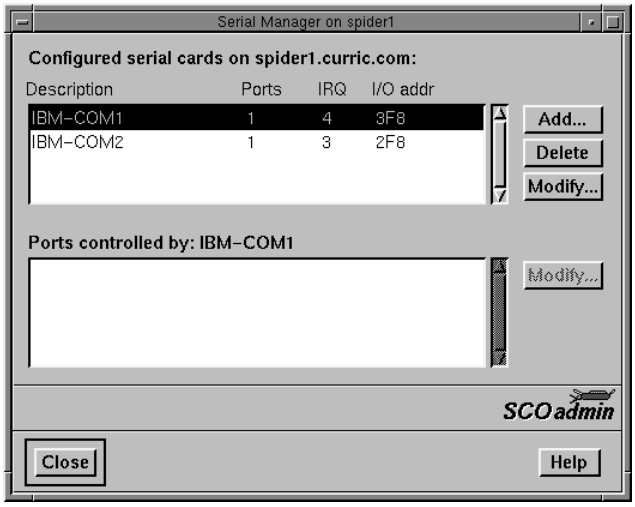
Step 4. To configure a modem manually, select **Modem ⇨ Add ⇨ Manual configuration** and enter the following information:

Modem vendor	If listed, select the manufacturer. Otherwise, select "Standard Modem Types" which works for most Hayes-compatible modems.
Modem model	If listed, select you modem model. Otherwise select the "Standard Modem" for the speed of your modem.
Line speed	The speed set in this field must match the speed set later in the *Systems* file. Note that this is not the speed at which the modems communicate. It is the speed at which the serial port and the modem communicate. Set it as high as possible for the type of hardware you have. (See Step 7). The default is 38400.
Modem port	Select the modem control serial device (such as tty2A) corresponding to the port the modem is connected to.
Answer mode	Select "on" if configuring incoming PPP connections. Otherwise select "Off" to prevent your system from accepting calls.

Step 5. Click on **OK** to accept the configuration.

AV 6-13

Serial Manager



NADM0613 © The Santa Cruz Operation, Inc.

AV 6-14

/etc/inittab Entries

Modem on COM1 configured to answer incoming calls:

```
Se1a:234:off:/etc/getty  tty1a m
Se1A:234:respawn:/etc/getty  -t60  tty1A  o
```

Modem on COM2 configured for outgoing calls only:

```
Se2a:234:off:/etc/getty  tty1a m
Se2A:234:off:/etc/getty  -t60  tty1A  o
```

NADM0614 © The Santa Cruz Operation, Inc.

AV 6-13: The Serial Manager
This is a screen shot of the **Serial Manager**.

Step 6. Modify the serial port. Click on **Configure port**. Selecting this option starts the **Serial Manager**, shown in AV 6-13.

Step 7. Select the serial port used for the modem, and click on **Modify.**

When configuring dial out connections, you might not need to make any modifications. For dial in connections, use these guidelines:

Speed(bps)	Try the highest speed port speed supported by your hardware. Match this speed setting with the modem configuration (Step 4). For example, if you set the modem speed to 38400, use that setting here. The setting depends on your hardware:
	8250 UART Select no higher than 9600 bps
	16450 UART Select no higher than 19200 bps
	16550 UART Set up to 115200 bps
Line-mode label	Leave as is unless you are debugging problems with an incoming connection.
Login/answer	Select “On” for dial in access. This enables the port for login.
Receive buffer	Defer modifying this control until you need to debug problems or improve performance.

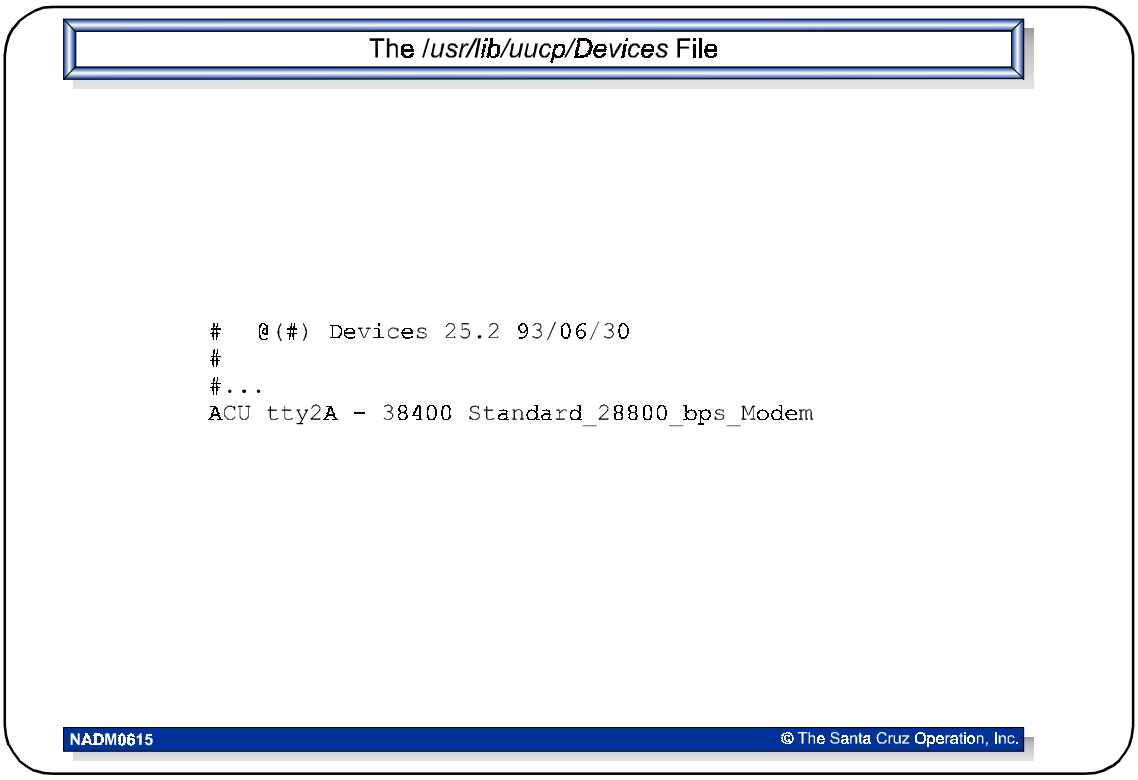
Step 8. Click on **OK** and then **Close** to exit the **Serial Manager**.

You see a message that the kernel environment is being rebuilt and these files are updated:

/etc/conf/init.d/sio	See the examples in AV 6-14
/etc/inittab	Updated from */etc/conf/init.d/sio*

AV 6-14: */etc/inittab* Entries
This slide shows an example */etc/inittab* file after the **Serial Manager** has configured the port for dial-out or dial-in connections. NOTE: Warn students against editing */etc/inittab* to manually configure the port.

AV 6-15


The /usr/lib/uucp/Devices File
@(#) Devices 25.2 93/06/30
#
#...
ACU tty2A - 38400 Standard_28800_bps_Modem
NADM0615
© The Santa Cruz Operation, Inc.

AV 6-15: The */usr/lib/uucp/Devices* File
This slide shows an example */usr/lib/uucp/Devices* file modified using the **Modem Manager**.

Step 9. Select **Host** ⇨ **Exit** to quit the **Modem Manager**.

When complete, the **Modem Manager** creates entries in */usr/lib/uucp/Devices* similar to those shown in AV 6-15. The format of the *Devices*(F) file is:

Connection type	Specifies one of two keywords: • Direct—connects to the modem; useful for testing, debugging, and setting modem configuration • ACU—specifies that this is a calling unit
Line	Device name of the modem control port
Dialer line	Used only with 801 dialers; enter a hyphen as a placeholder
Speed	The speed or speed range for the device—matched against the speed field of the entries in */usr/lib/uucp/Systems* with ACU as the device field.
Dialer	Specifies the name of the dialer program (for ACU entries) or direct. Dialer programs must be present in */usr/lib/uucp*.

Exercise: Add a modem

Run the **Modem Manager** and configure the modem for dialout connections.

Use **more**(C) to view */usr/lib/uucp/Devices* and */etc/inittab* modified by the **Modem Manager**.

6.4.1.1 Testing the modem

To test a dial out connection, use **cu**(C) with a phone number supplied by your instructor. For example:

```
cu 5551212
```

If you have problems, enable debugging with the **-x9 flag**:

```
cu -x9 5551212
```

Enter **~.** to drop a connection after it is established.

To test a dial-in configuration, use **cu** from a remote UNIX system, or use a Windows terminal emulation program, such as **HyperTerminal** to call the modem.

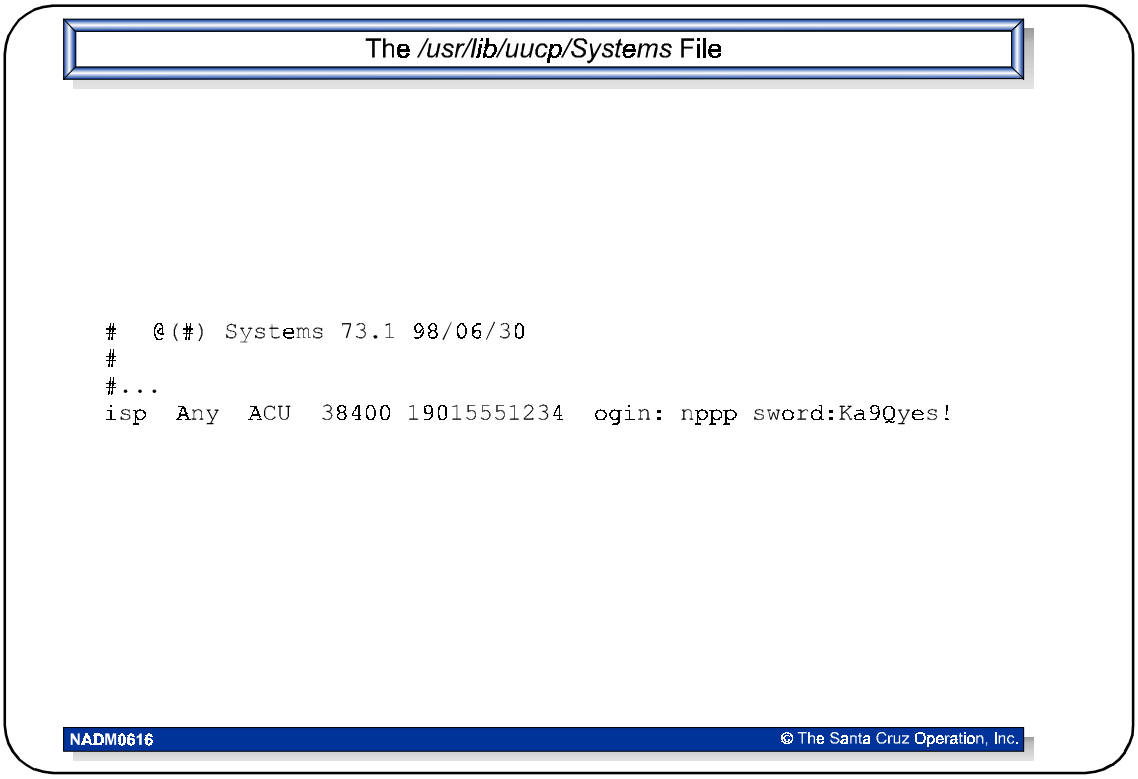
If **cu** fails, see Section 6.6, “Troubleshooting PPP,” on page 6-80. Also refer to **scohelp ⇨ SCO OpenServer Handbook ⇨ Troubleshooting modems**.

Exercise: Test the modem

Test the modem with **cu** using the phone number provided by your instructor:

```
cu -x9 ###-####
```

AV 6-16


The /usr/lib/uucp/Systems File
@(#) Systems 73.1 98/06/30
#
#...
isp Any ACU 38400 19015551234 ogin: nppp sword:Ka9Qyes!
NADM0616
© The Santa Cruz Operation, Inc.

6.4.2 Configuring UUCP

AV 6-16: The */usr/lib/uucp/Systems* File
This slide shows an example */usr/lib/uucp/Systems* file.

After you have configured */usr/lib/uucp/Devices* and verified that you can dial out from your system using **cu**(C), edit the */usr/lib/uucp/Systems* file to configure information about the dial-out connection to the remote system. Note that the *Systems*(F) file is not used for incoming connections.

Add a line similar to the one shown in AV 6-16. The specific entries you will need depend on your port and modem attributes. The fields in the *Systems* file are:

Site name	Name by which the remote system is known. This does not have to be the actual remote system name. It is the name by which that system is known locally.
Schedule	Day/time when you may call. *Any* day is commonly used for dialout connections.
Device type	Specifies one of two keywords, matching the corresponding field in */usr/lib/uucp/Devices*: • Direct—connects to the modem • ACU—specifies that this is a calling unit
Speed	The speed or speed range for the device. This field is matched against the corresponding field in */usr/lib/uucp/Devices*.
Phone	Phone number for the remote system
Login script	Specifies the login sequence on the remote system in the format: `expect send` In this example, when the string "`ogin`" is received, the login name "`nppp`" is returned.

When you have added an entry for your outgoing connection, test the connection with **cu**:

```
cu site-name
```

The modem will dial the number in the *Systems* file that corresponds to the entry beginning with `site-name.`

Note that the login script information entered in the *Systems* file is not used with **cu**. It is used when connecting with PPP. To complete this test connection, you would enter your login information manually.

To test the login script, use the **uutry**(ADM) command. As with **cu**, the **-x9** option sets debugging:

```
/usr/lib/uucp/uutry -x9 -r site-name
```

Exercise: Configure and test UUCP

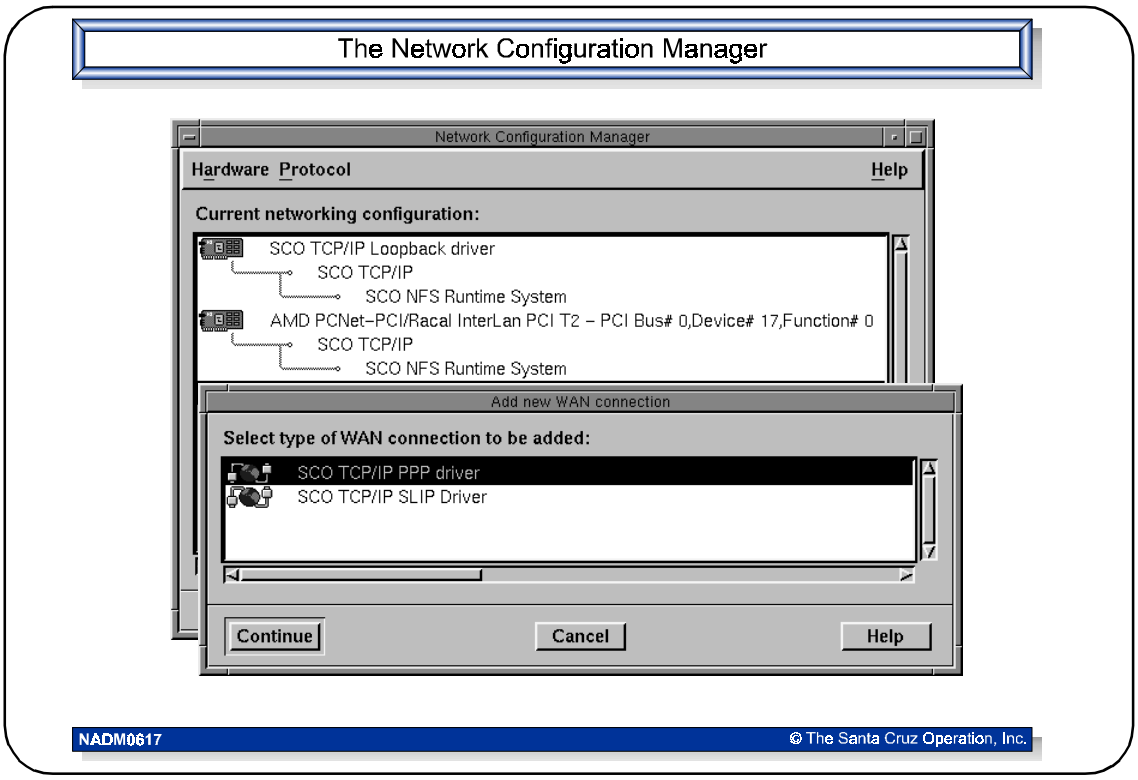
Create the file */usr/lib/uucp/Systems* and add an ACU entry for your remote system using information provided by your instructor.

Test the configuration:

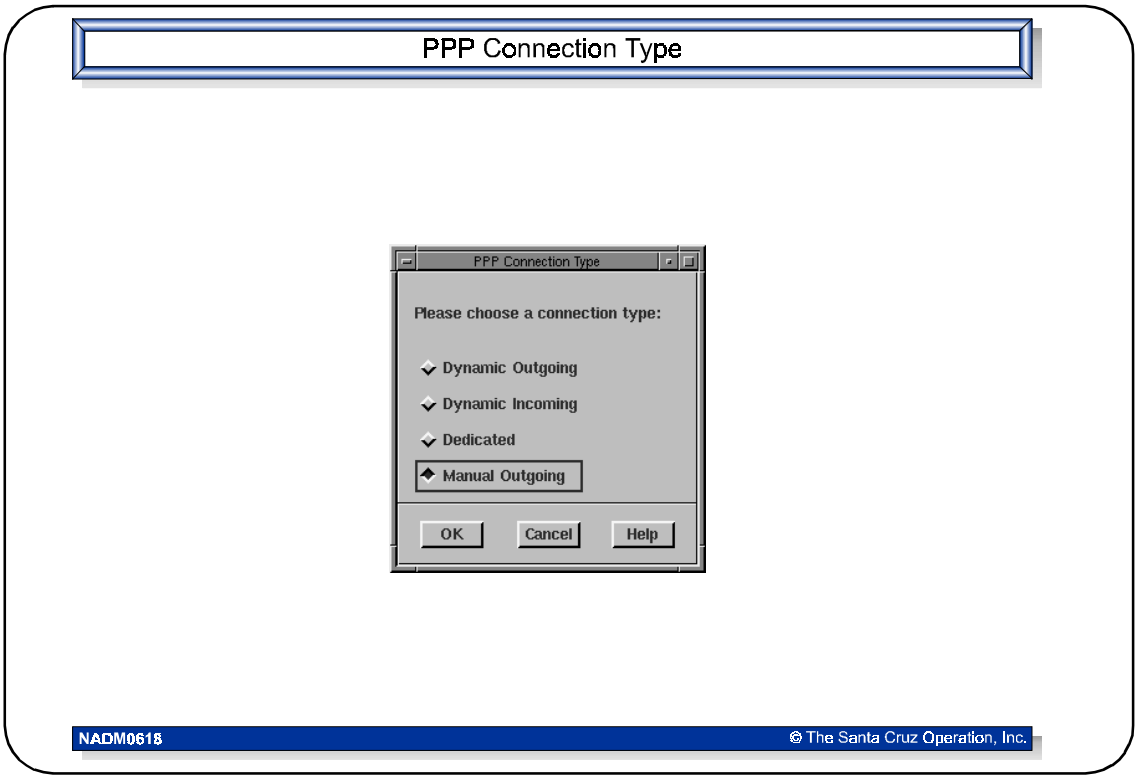
```
cu site-name
```

```
/usr/lib/uucp/uutry -x9 -r site-name
```

AV 6-17


The Network Configuration Manager
Network Configuration Manager
Hardware Protocol
Help
Current networking configuration:
SCO TCP/IP Loopback driver
SCO TCP/IP
SCO NFS Runtime System
AMD PCNet–PCI/Racal InterLan PCI T2 – PCI Bus# 0,Device# 17,Function# 0
SCO TCP/IP
SCO NFS Runtime System
Add new WAN connection
Select type of WAN connection to be added:
SCO TCP/IP PPP driver
SCO TCP/IP SLIP Driver
Continue
Cancel
Help
NADM0617
© The Santa Cruz Operation, Inc.

AV 6-18


PPP Connection Type
PPP Connection Type
Please choose a connection type:
Dynamic Outgoing
Dynamic Incoming
Dedicated
Manual Outgoing
OK
Cancel
Help
NADM0618
© The Santa Cruz Operation, Inc.

6.4.3 Configuring SCO PPP outgoing connections

The **Network Configuration Manager** configures SCO PPP for an outgoing connection by performing these tasks:

- Adding the PPP driver
- Configuring the TCP/IP protocol
- Modifying the PPP configuration file, */etc/ppphosts*
- Modifying other PPP configuration files (if needed), such as */etc/pppauth*
- Modifying */etc/tcp* to start the PPP daemon, **pppd,** at boot time

When finished, the **Network Configuration Manager** prompts you to relink the kernel and reboot the system.

6.4.3.1 Configuring an SCO PPP manual outgoing connection

AV 6-17: Network Configuration Manager
This slide is a screen shot of the selection **Hardware ⇨ Add a new WAN Connection** in the **Network Configuration Manager.**

To configure a manual outgoing PPP connection:

Step 1. Configure the hardware and UUCP. See Section 6.4.1 on page 6-29 and Section 6.4.2 on page 6-39.

Step 2. Start the **Network Configuration Manager**

Step 3. Select **Hardware ⇨ Add new WAN connection.**

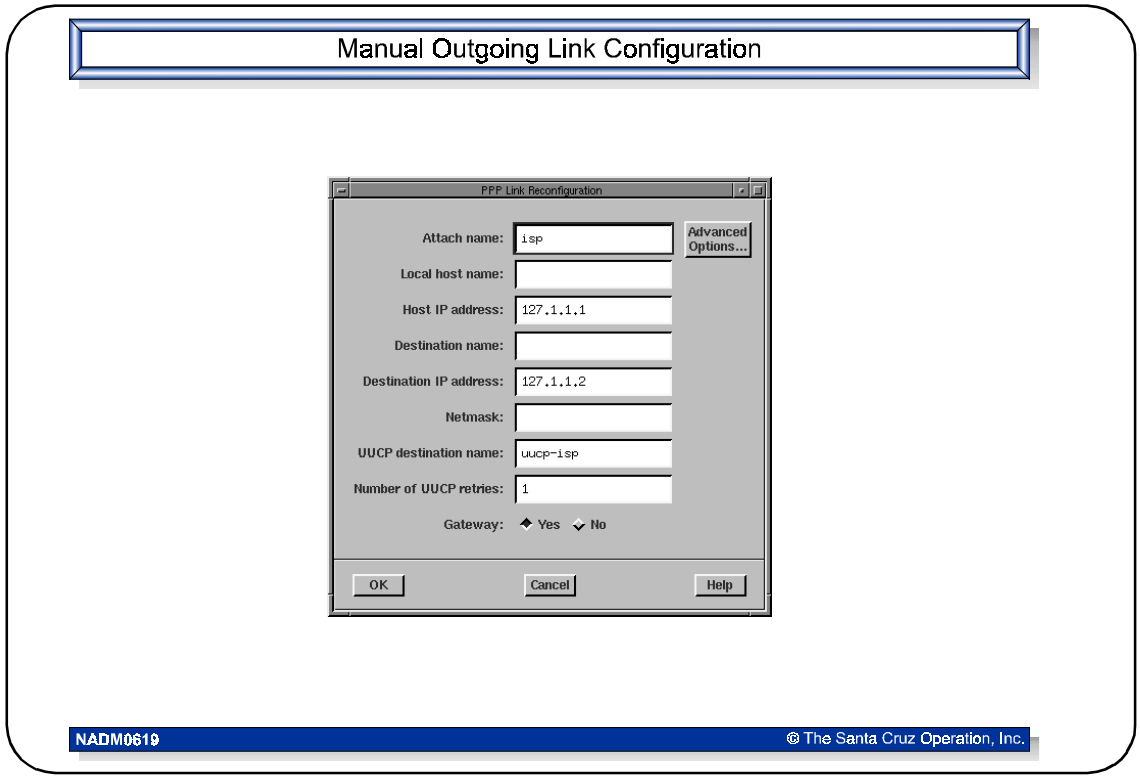
Step 4. Select “SCO TCP/IP PPP driver” from this list of WAN connection types and click on **Continue**.

Step 5. Select “SCO TCP/IP” as the networking product to be added to the PPP driver and click on **Add**.

Step 6. Choose “Manual Outgoing” as the connection type and click on **OK**.

AV 6-18: PPP Connection Type
This slide is a screen shot of the **PPP Connection Type** selection window**.**

AV 6-19


Manual Outgoing Link Configuration
PPP Link Reconfiguration
Attach name:
isp
Advanced Options...
Local host name:
Host IP address:
127.1.1.1
Destination name:
Destination IP address:
127.1.1.2
Netmask:
UUCP destination name:
uucp-isp
Number of UUCP retries:
1
Gateway:
Yes
No
OK
Cancel
Help
NADM0619
© The Santa Cruz Operation, Inc.

AV 6-19: Manual Outgoing Link Configuration
This slide shows the configuration form for a manual outgoing link. In this configuration, both the local and remote IP addresses are assigned by the remote host. Note that when names are not entered, the /etc/hosts file is not changed.

Step 7. Fill in the configuration form:

Attach name — This is the name that is used to bring up the connection with the **pppattach** command. For example, if you enter 'isp' here, you will bring up the outgoing connection with the command:

```
pppattach isp
```

Local host name — Enter a unique name by which this interface is known locally. The name is added to */etc/hosts* with the IP address entered in the next field.

If the remote host assigns the IP address, leave the local host name blank.

Host IP address — Enter an IP address for the local-side of the connection. It is added to */etc/hosts* if a local name is entered in the previous field.

The remote system ignores this address if it assigns the IP address.

Destination name — The name by which the remote system is known. This does not need to match the host name configured on the remote system. The name is added to */etc/hosts* with the destination IP address entered in the next field.

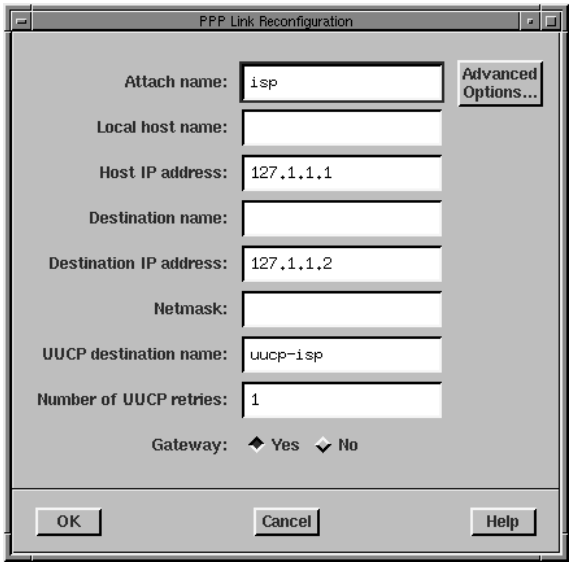
If the remote host assigns the IP address, leave the destination host name blank.

Destination IP address — Enter a unique IP address for the remote-side of the connection. It is added to */etc/hosts* if a destination name is entered in the previous field.

The remote system ignores this address if it assigns the IP address.

AV 6-20

Manual Outgoing Link



/etc/ppphosts

```
127.1.1.1:127.1.1.2 attach=isp uucp=uucp-isp retry=2
```

/usr/lib/uucp/Systems

```
uucp-isp  Any  ACU  38400  nnnnnnn  login:nppp sword:Ka9yes!
```

/usr/lib/uucp/Devices

```
ACU tty1A - 38400 Standard_28800_bps_Modem
```

NADM0620 © The Santa Cruz Operation, Inc.

Netmask — If specifying IP addresses, enter the netmask. The netmask must match the netmask used by the remote system. The default is 255.255.255.0.

If using dynamic IP addresses assigned by the remote server, leave this blank.

UUCP destination name — Enter the site name of the remote system configured in */usr/lib/uucp/Systems*.

Number of UUCP retires — Specify the number of times UUCP tries to connect to the remote system. The default is 2.

Gateway — Select "Yes" if this PPP interface is not the first on your system and you want to route packets between them.

Step 8. Click on **OK** when the form is filled in.

Step 9. Click on **No** when asked to configure more links.

Step 10. Click on **OK** when the successful configuration message displays.

Step 11. Select **Hardware ⇨ Exit.**

Step 12. Click on **Yes** to relink the Unix kernel. Answer "y" to the prompts to boot the new kernel by default and rebuild the kernel environment.

Step 13. Press ENTER to continue when the kernel has been rebuilt, and exit the **Network Configuration Manager**.

Step 14. Reboot the system.

When the system is restarted, the PPP daemon is started and reads the new entry created in */etc/ppphosts*, as shown in AV 6-20. Note that the UUCP files are not modified by the **Network Configuration Manager**. They are included to show the relationships between the UUCP and PPP configuration files.

AV 6-20: Manual Outgoing Link
This slide shows the relationship between the entries in **netconfig**(ADM) and the */etc/ppphosts* file that is created. It also shows the corresponding entries in */usr/lib/uucp/Systems* and */usr/lib/uucp/Devices*.

6.4.3.2 Initiating a manual outgoing PPP connection

Once PPP has been configured, connect using a manual link with **pppattach**(ADMN):

pppattach *site-name*

When the connection succeeds, use **netstat**(TC) to check the PPP interface. If the connection is configured to use remotely-assigned IP addresses, **netstat** will display the assigned IP address:

```
netstat -in
Name   Mtu    Network      Address         Ipkts Ierrs Opkts Oerrs Coll
net0   1500   150.200.76   150.200.76.1     5008  0     1495  0     0
lo0    8232   127          127.0.0.1       820   0     820   0     0
atl0*  8232   none         none            No Statistics Available
ppp0   1500   207.143.209  207.143.209.82  15    0     16    0     0
```

Test the interface using **ping**(ADMN). For example:

```
ping 207.142.209.82
```

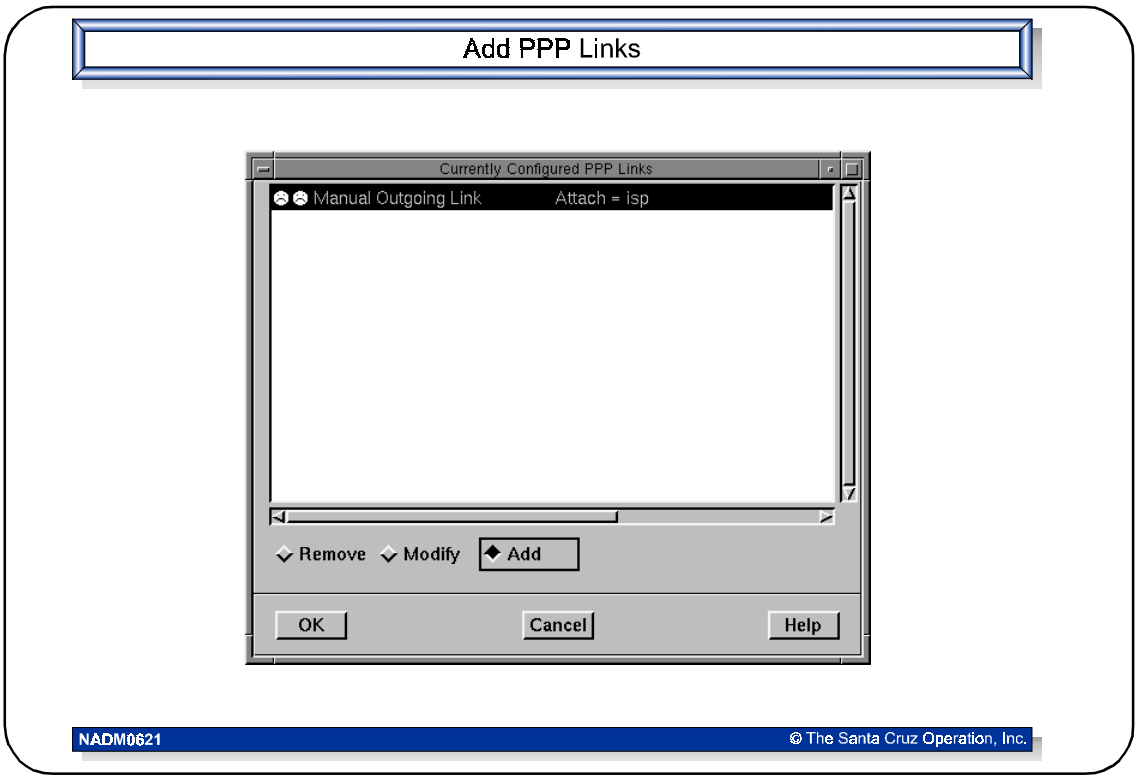
If this PPP link is the default route to the Internet, add the IP address to the routing table using **route**(ADMN). For example:

```
route add default 207.142.209.82
```

Instructor Guide

This would be a good time to do the lab in Section 6.8.5, “Configuring a modem for incoming calls,” on page 6-92. If students completed the exercises for configuring the modem and the Systems file, you can bypass the labs in Section 6.8.2, “Configuring a modem for outgoing calls,” on page 6-88 and Section 6.8.3, “Configuring /usr/lib/uucp/Systems,” on page 6-89.

AV 6-21


Add PPP Links
Currently Configured PPP Links
Manual Outgoing Link
Attach = isp
Remove
Modify
Add
OK
Cancel
Help
NADM0621
© The Santa Cruz Operation, Inc.

6.4.3.3 Configuring an SCO PPP dynamic outgoing connection

The procedure for configuring a dynamic outgoing connection is similar to configuring a manual connection. To configure a dynamic outgoing connection:

Step 1. Configure and test the modem, if you have not already done so. (See Section 6.4.1 on page 6-29.)

Step 2. Configure UUCP for this connection by making a new entry in */usr/lib/uucp/Systems* for this site. (See Section 6.4.2 on page 6-39.)

Step 3. If you have not yet configured PPP, follow the procedures in Section 6.4.3.1, "Configuring an SCO PPP manual outgoing connection," on page 6-43 noting these differences:

- In Step 6, select "Dynamic outgoing" as the PPP connection type.
- In Step 7, note the differences in the link configuration parameters when configuring a Dynamic link. Follow the guidelines below in Step 9.

Step 4. If PPP is already configured, start the **Network Configuration Manager** as **root**.

Step 5. Select "SCO TCP/IP" under "SCO TCP/IP PPP driver."

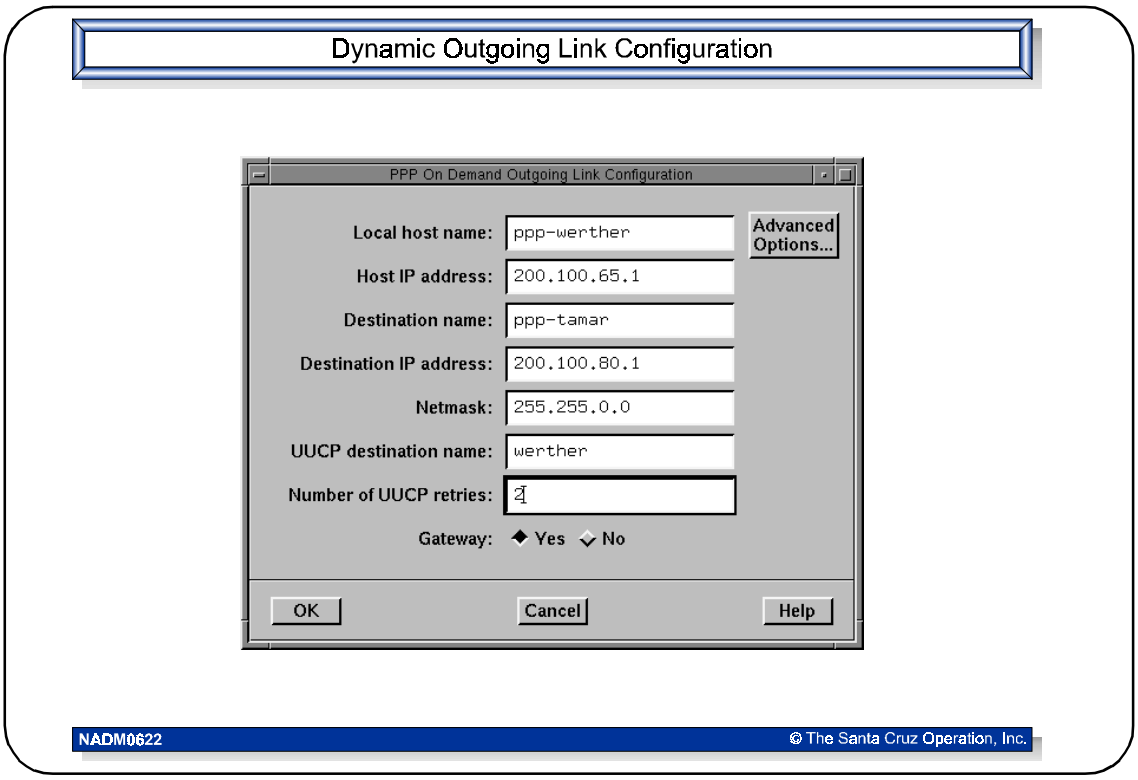
Step 6. Select **Protocol ⇨ Modify protocol configuration.**

Step 7. Toggle "Add" and click on **OK** when the currently configured links are displayed.

Step 8. Toggle "Dynamic Incoming" and click on **OK**.

AV 6-21: Add PPP Links
This slide shows the **Network Configuration Manager** selection **Protocol ⇨ Modify protocol configuration**, used to add PPP endpoints (links).

AV 6-22


Dynamic Outgoing Link Configuration
PPP On Demand Outgoing Link Configuration
Local host name:
ppp-werther
Advanced Options...
Host IP address:
200.100.65.1
Destination name:
ppp-tamar
Destination IP address:
200.100.80.1
Netmask:
255.255.0.0
UUCP destination name:
werther
Number of UUCP retries:
2
Gateway:
Yes
No
OK
Cancel
Help
NADM0622
© The Santa Cruz Operation, Inc.

AV 6-22: Dynamic Outgoing Link Configuration
This slide shows the screen for configuring a dynamic outgoing link.

Step 9. Fill in the dynamic outgoing link configuration form shown in AV 6-22:

Local host name — Enter a unique name by which this interface is known locally. The name is added to */etc/hosts* with the IP address entered in the next field.

If the remote host assigns the IP address, leave the local host name blank.

Host IP address — Enter an IP address for the local side of the connection. It is added to */etc/hosts* if a local name is entered in the previous field.

The remote system will ignore this address if it assigns the IP address.

Destination name — The name by which the remote system is known. This does not need to match the host name configured on the remote system. This is added to */etc/hosts* with the destination IP address entered in the next field.

If the remote host assigns the IP address, leave the destination host name blank.

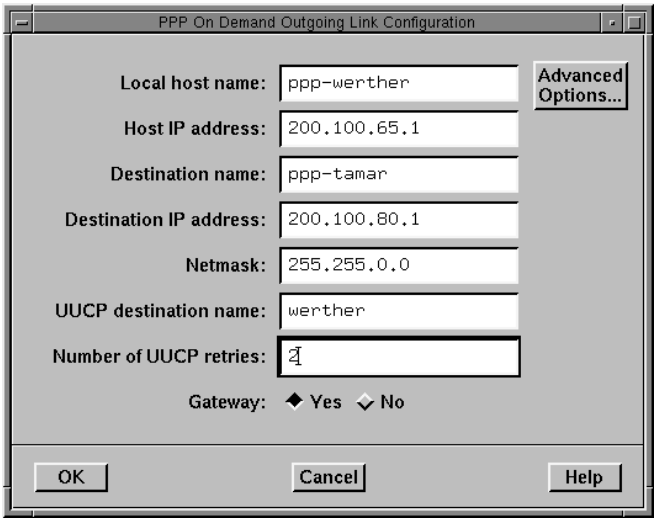
Destination IP address — Enter a unique IP address for the remote side of the connection. It is added to */etc/hosts* with the destination name.

The remote system will ignore this address if it assigns the IP address.

If both a name and IP address are specified now, the IP address shows up as 0.0.0.0 when **netconfig**(ADM) is used to modify this entry in the future. This is because only the name is saved in */etc/ppphosts* when both the name and IP address are specified. Although it may appear that this must be changed to a 'real' IP address, this is not the case. The */etc/hosts* file correctly maps the name to the IP address originally entered.

AV 6-23

Dynamic Outgoing Link



/etc/ppphosts

```
ppp-tamar:ppp-werther uucp=werther retry=2 mask=255.255.0.0
```

/etc/hosts

```
200.100.65.1     ppp-werther
200.100.65.2     ppp-tamar
```

NADM0623 © The Santa Cruz Operation, Inc.

Netmask — If specifying IP addresses, enter the netmask. The netmask must match that used by the remote system. The default is 255.255.255.0.

If using dynamic IP addresses assigned by the remote server, leave this blank.

UUCP destination name — Enter the site name of the remote system configured in */usr/lib/uucp/Systems*.

Number of UUCP retires — Specify the number of times UUCP tries to connect to the remote system.

Gateway — Select "Yes" if this PPP interface is not the first on your system, and you want to route packets between them.

Step 10. Click on **OK** when the form is filled in.

Step 11. Click on **No** when asked to configure more links.

Step 12. Click on **OK** when the successful configuration message displays.

Step 13. Select **Hardware ⇨ Exit.**

If you had previously configured the PPP driver, you are not prompted to relink the kernel.

A new entry is added to */etc/ppphosts* for the dynamic outgoing link and the host names and IP addresses are added to */etc/hosts*. AV 6-23 shows an example entry.

AV 6-23: Dynamic Outgoing Link
This slide shows how the entries in **netconfig**(ADM) are used to configure */etc/ppphosts* and */etc/hosts*.

6.4.3.4 Initiating a dynamic outgoing PPP connection

Dynamic outgoing links are established when an IP packet is detected for the remote site. To test the dynamic link configuration, use **ping**(ADMN) or **telnet**(TC) to send a packet to the remote site:

```
ping 200.100.65.1
```

Use **netstat**(TC) to verify that the interface is up. You should see an entry for the PPP interface. For example:

```
netstat -in
ppp0   1500   200.100      200.100.65.1   15    0     16    0     0
```

If this PPP link is the default route to the Internet, add the IP address to the routing table using **route**(ADMN). For example:

```
route add default 200.100.65.1
```

To close a PPP connection, use the **ifconfig**(ADMN) command. For example:

```
ifconfig ppp0 down
```

6.4.4 Configuring SCO PPP incoming connections

Configuring incoming PPP connections is similar to the procedure for creating outgoing connections, except the UUCP *Systems* file (*/usr/lib/uucp/Systems*) does not need to be configured.

To configure an incoming PPP connection:

Step 1. Configure the modem to answer calls (see Section 6.4.1 on page 6-29).

Step 2. If you do not already have the PPP driver configured, follow the procedure in Section 6.4.3.1 on page 6-43, noting these differences:

- In Step 6, select “Dynamic incoming” as the PPP connection type.
- In Step 7, note the differences in the link configuration parameters when configuring a Dynamic link. Follow the guidelines for Step 8 below.

Step 3. If PPP is already configured, start the **Network Configuration Manager** as **root**.

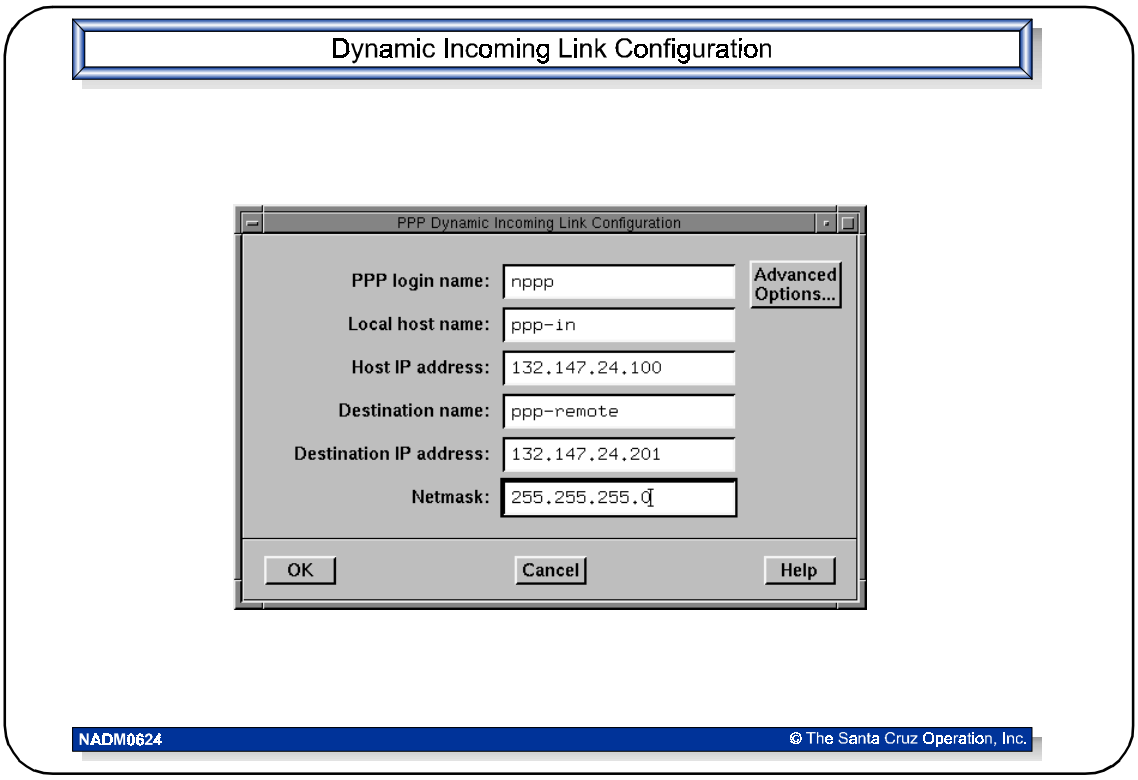
Step 4. Select “SCO TCP/IP” under “SCO TCP/IP PPP driver.”

Step 5. Select **Protocol ⇨ Modify protocol configuration.**

Step 6. Toggle “Add” and click on **OK** when the currently configured links are displayed.

Step 7. Toggle “Dynamic Incoming” and click on **OK**.

AV 6-24



Instructor Guide

AV 6-24: Dynamic Incoming Link Configuration
This slide shows the screen for configuring a dynamic outgoing link.

Discuss various login name schemes with your students. For example, it is possible to set up each remote machine to use a different login name for establishing PPP connections. This can be very useful for security auditing and PPP troubleshooting. Or, each remote machine could just use the same account name.

Step 8. Fill in the form shown in AV 6-24 using these guidelines and click on **OK** when finished.

PPP login name: Accept the default of **nppp** or change the name. A user account with this name is added to */etc/passwd* with a login shell of */usr/lib/ppp/ppp*, which passes connection information to **pppd**.

Local host name: Enter a name for this incoming link. This name must be different from any other networking interface configured on this system.

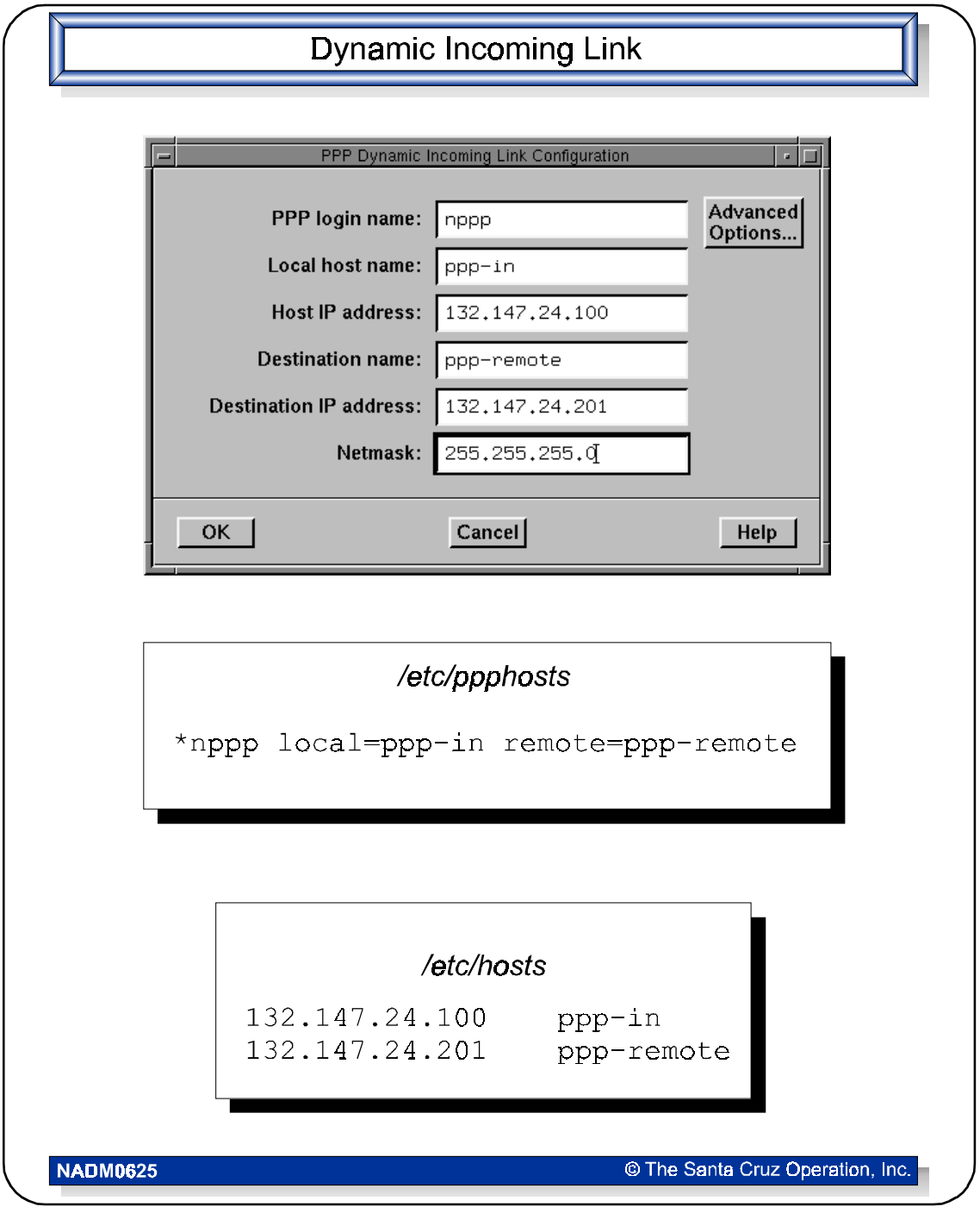
Host IP address: There are three options for assigning IP addresses for the local side of an incoming link:

- Enter a unique IP address for this connection.
- Leave this field blank if you want the remote system to assign an IP address.
- Use */etc/ppppool* to dynamically assign an address from a pool.

It is recommended that you specify an IP address initially to verify that the connection works properly in a basic configuration.

Destination name: Enter a name for the remote system. The choice of name is arbitrary (but must be unique on this system).

AV 6-25


Dynamic Incoming Link
PPP Dynamic Incoming Link Configuration
PPP login name:
nppp
Advanced Options...
Local host name:
ppp-in
Host IP address:
132.147.24.100
Destination name:
ppp-remote
Destination IP address:
132.147.24.201
Netmask:
255.255.255.0
OK
Cancel
Help
/etc/ppphosts
*nppp local=ppp-in remote=ppp-remote
/etc/hosts
132.147.24.100 ppp-in
132.147.24.201 ppp-remote
NADM0625
© The Santa Cruz Operation, Inc.

Destination IP address

As with local addresses, there are three options for assigning IP addresses for the remote side of an incoming link:

- Enter a unique IP address for this connection.
- Leave this field blank and the remote system assigns an IP address.
- Use */etc/ppppool* to dynamically assign an address from a pool.

It is recommended that you specify an IP address initially to verify that the connection works properly in a basic configuration.

Netmask — Enter the netmask. The netmask must match that used by the remote system.

The user account is now created using the account name provided in the configuration form.

Step 9. Type “y” or “n” when prompted to specify a user ID (UID) and enter a UID number. If you do not specify a UID, the system assigns one. Press ENTER to continue.

Step 10. Enter the password for the new PPP login account when prompted.

Step 11. Click on **No** when asked to configure more links.

Step 12. Click on **OK** when the successful configuration message displays,

Step 13. Select **Hardware ⇨ Exit.** You are not prompted to relink the kernel if the PPP driver is already configured.

When finished, a new entry is added to */etc/ppphosts* for the dynamic incoming link. AV 6-25 shows an example entry.

A new user account is created with the login and password you specified, and a login shell of */usr/lib/ppp/ppp.*

AV 6-25: /etc/ppphosts—Dynamic Incoming Link
This slide shows an example entry in /etc/ppphosts for a dynamic incoming link.

6.4.5 Configuring SCO PPP dedicated connections

Dedicated connections are used with a serial line that is reserved exclusively for a single connection, such as a leased line between buildings. Either host can initiate the connection.

As with other types of PPP connections, the **Network Configuration Manager** is used to configure dedicated connections.

Step 1. Configure the serial port, if necessary.

Step 2. If you have not yet configured PPP, follow the procedures in Section 6.4.3.1, "Configuring an SCO PPP manual outgoing connection," on page 6-43 noting these differences:

In Step 6, select "Dedicated" as the PPP connection type.

In Step 7, note the differences in the link configuration parameters when configuring a dedicated link. Follow the guidelines for Step 8 below.

Step 3. If PPP is already configured, start the **Network Configuration Manager** as **root**.

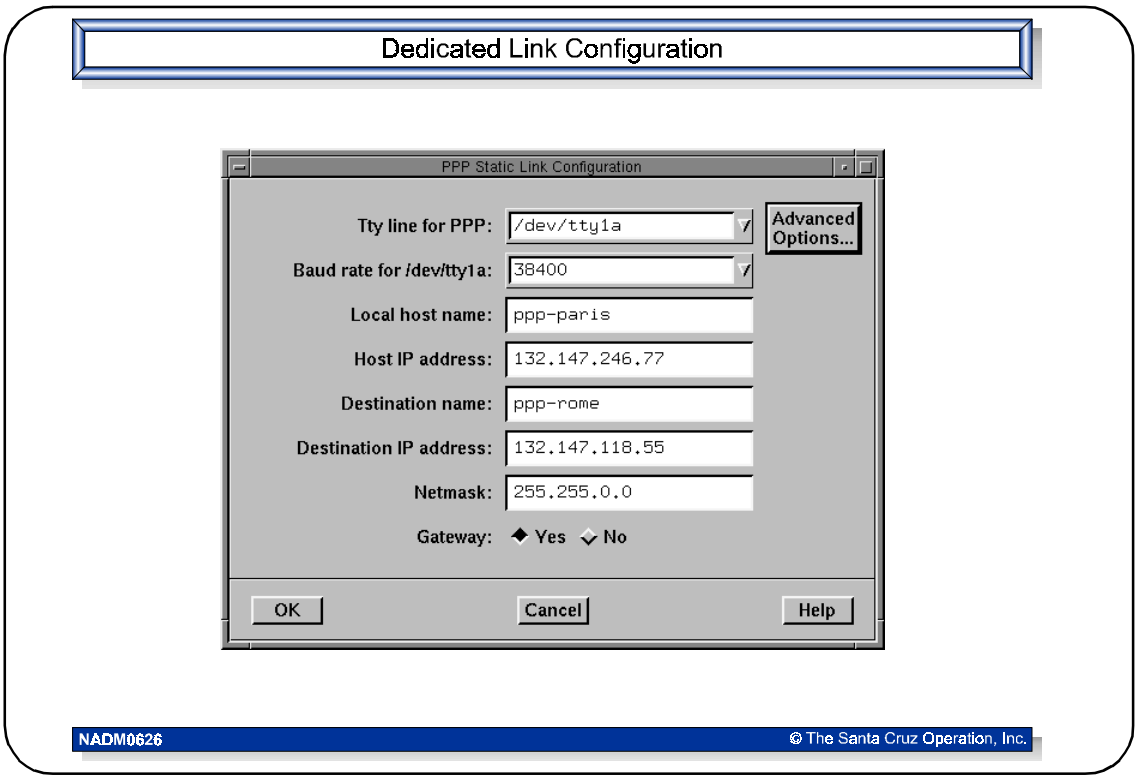
Step 4. Select "SCO TCP/IP" under "SCO TCP/IP PPP driver."

Step 5. Select **Protocol ⇨ Modify protocol configuration**.

Step 6. Toggle "Add" and click on **OK** when the currently configured links are displayed.

Step 7. Toggle "Dedicated" and click **OK**.

AV 6-26

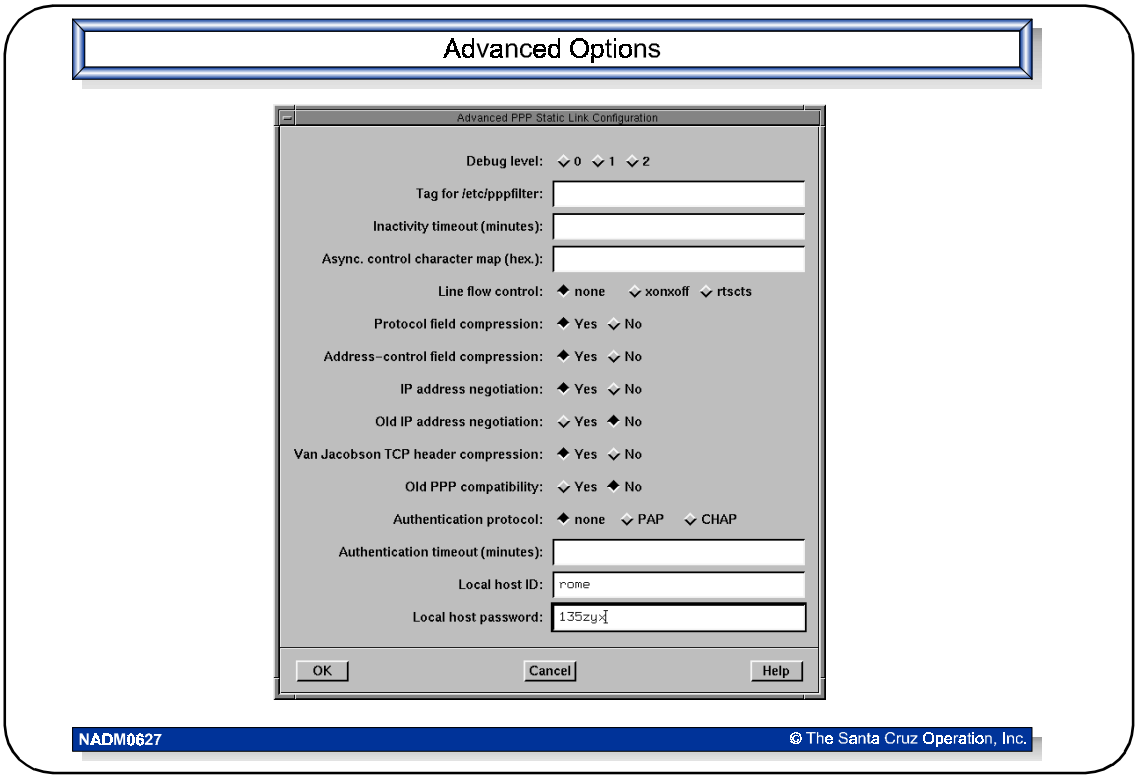

Dedicated Link Configuration
PPP Static Link Configuration
Tty line for PPP:
/dev/tty1a
Advanced Options...
Baud rate for /dev/tty1a:
38400
Local host name:
ppp-paris
Host IP address:
132.147.246.77
Destination name:
ppp-rome
Destination IP address:
132.147.118.55
Netmask:
255.255.0.0
Gateway:
Yes
No
OK
Cancel
Help
NADM0626
© The Santa Cruz Operation, Inc.

AV 6-26: Dedicated Link Configuration
This slide shows the screen for configuring a dedicated link.

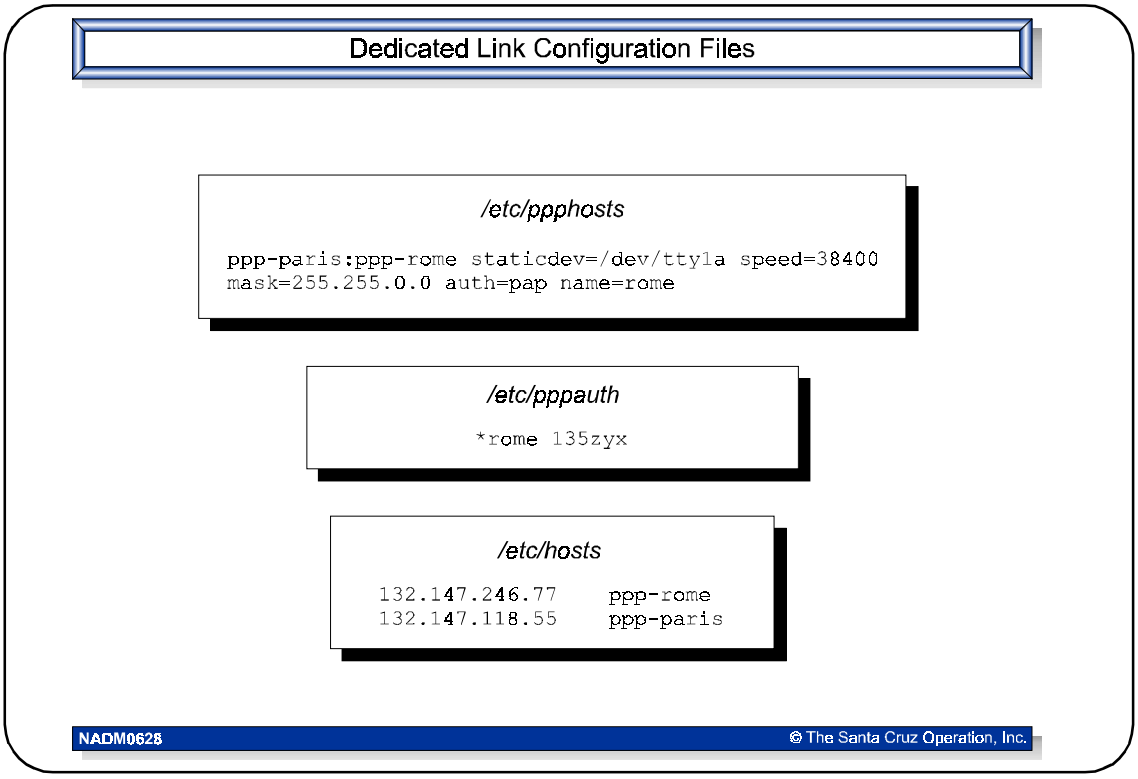
Step 8. Fill in the form shown in AV 6-26 using these guidelines:

Tty line for PPP	Enter the serial port used by this link. Use the non-modem-control port.
Baud rate	Enter the line speed for the serial port.
Local host name:	Enter a name by which this interface is known locally. NOTE: This name must be different from any other networking interface configured on this system.
Host IP address:	Enter a unique IP address for this connection. The remote system must agree to use the specified address or the local system refuses to establish the link. The host name and IP address are entered in */etc/hosts*.
Destination name	The name by which the remote system is known.
Destination IP address	Enter a unique IP address for the remote end of this PPP interface. The remote system must agree to use the specified address or the local system refuses to establish the link.
Netmask	Enter the netmask. The netmask must match that used by the remote system.
Gateway	If this network interface is not the first on your system and you want to route packets between them, select "Yes."

AV 6-27


Advanced Options
Advanced PPP Static Link Configuration
Debug level: 0 1 2
Tag for /etc/pppfilter:
Inactivity timeout (minutes):
Async. control character map (hex.):
Line flow control: none xonxoff rtscts
Protocol field compression: Yes No
Address-control field compression: Yes No
IP address negotiation: Yes No
Old IP address negotiation: Yes No
Van Jacobson TCP header compression: Yes No
Old PPP compatibility: Yes No
Authentication protocol: none PAP CHAP
Authentication timeout (minutes):
Local host ID: rome
Local host password: 135zyx
OK
Cancel
Help
NADM0627
© The Santa Cruz Operation, Inc.

AV 6-28


Dedicated Link Configuration Files
/etc/ppphosts
ppp-paris:ppp-rome staticdev=/dev/tty1a speed=38400
mask=255.255.0.0 auth=pap name=rome
/etc/pppauth
*rome 135zyx
/etc/hosts
132.147.246.77 ppp-rome
132.147.118.55 ppp-paris
NADM0628
© The Santa Cruz Operation, Inc.

AV 6-27: Advanced Options
This slide shows the **Advanced Options** form used to configure authentication for dedicated links.

Step 9. Click on **Advanced Options** to configure authentication. This is especially important for dedicated links, which use no other type of authentication (such as logging in) to access the system. This example uses Password Authentication Protocol (PAP):

Authentication protocol	Select PAP.
Local host ID	Enter a name for identifying the connection. This updates */etc/pppauth.*
Local host password	Enter a password used to authenticate the connection.

Step 10. Click on **OK** to exit the **Advanced Options** screen, and again to complete the PPP configuration.

Step 11. Click on **No** when asked to configure more links.

Step 12. Click on **OK,** when the successful configuration message displays,.

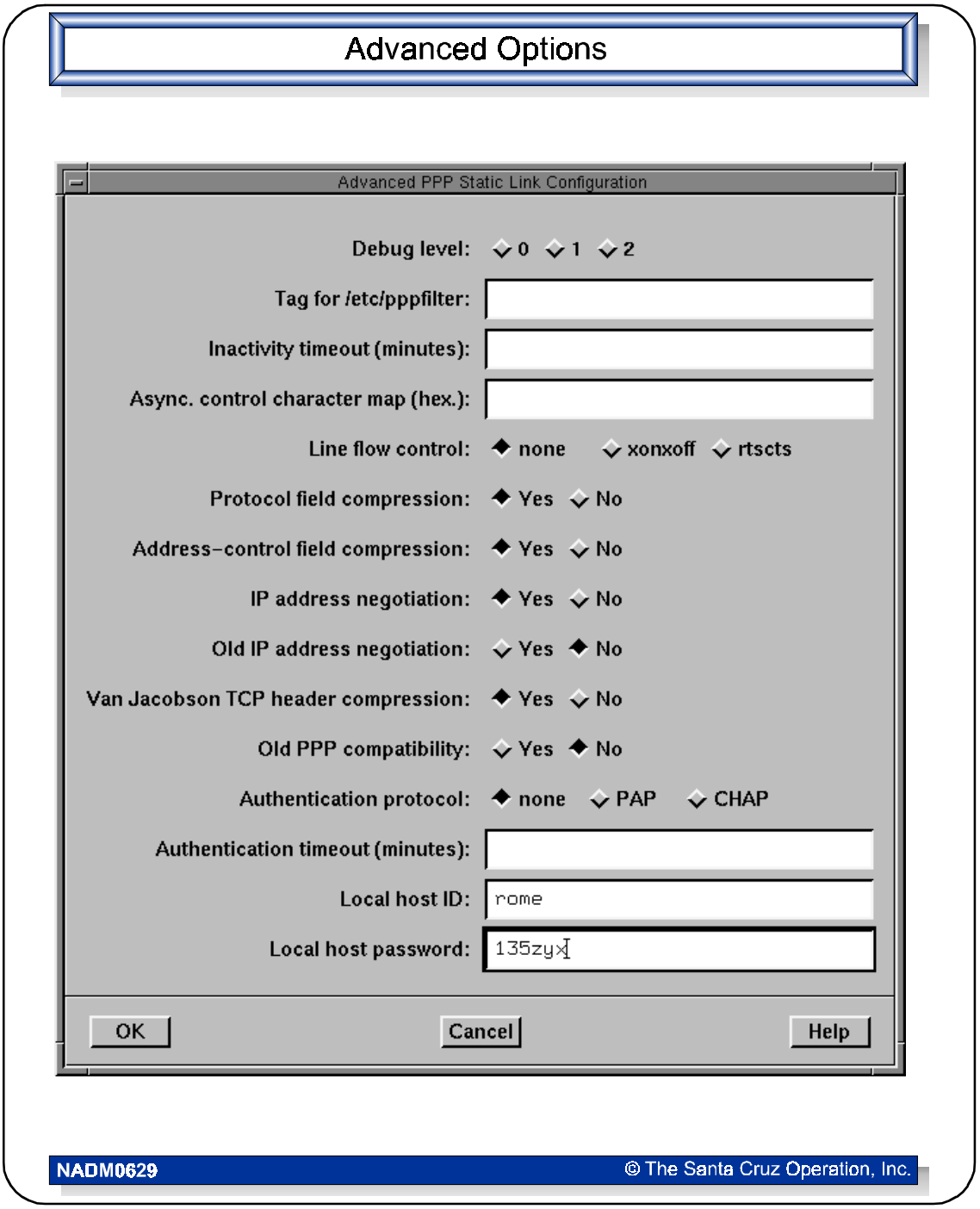
Step 13. Select **Hardware ⇨ Exit.**

You are not prompted to relink the kernel if the PPP driver is already configured.

A new entry is added to */etc/ppphosts* for the dynamic incoming link. The */etc/pppauth* file is updated with the PAP entries. AV 6-28 shows an example entry.

AV 6-28: Dedicated Link Configuration Files
This slide shows the PPP configuration entries for a dedicated link.

AV 6-29


Advanced Options
Advanced PPP Static Link Configuration
Debug level: 0 1 2
Tag for /etc/pppfilter:
Inactivity timeout (minutes):
Async. control character map (hex.):
Line flow control: none xonxoff rtscts
Protocol field compression: Yes No
Address-control field compression: Yes No
IP address negotiation: Yes No
Old IP address negotiation: Yes No
Van Jacobson TCP header compression: Yes No
Old PPP compatibility: Yes No
Authentication protocol: none PAP CHAP
Authentication timeout (minutes):
Local host ID: rome
Local host password: 135zyx
OK
Cancel
Help
NADM0629
© The Santa Cruz Operation, Inc.

6.4.6 Advanced SCO PPP configuration

AV 6-29: Advanced Options
This slide shows the **Advanced Options** form accessed from any of the PPP link configuration forms.

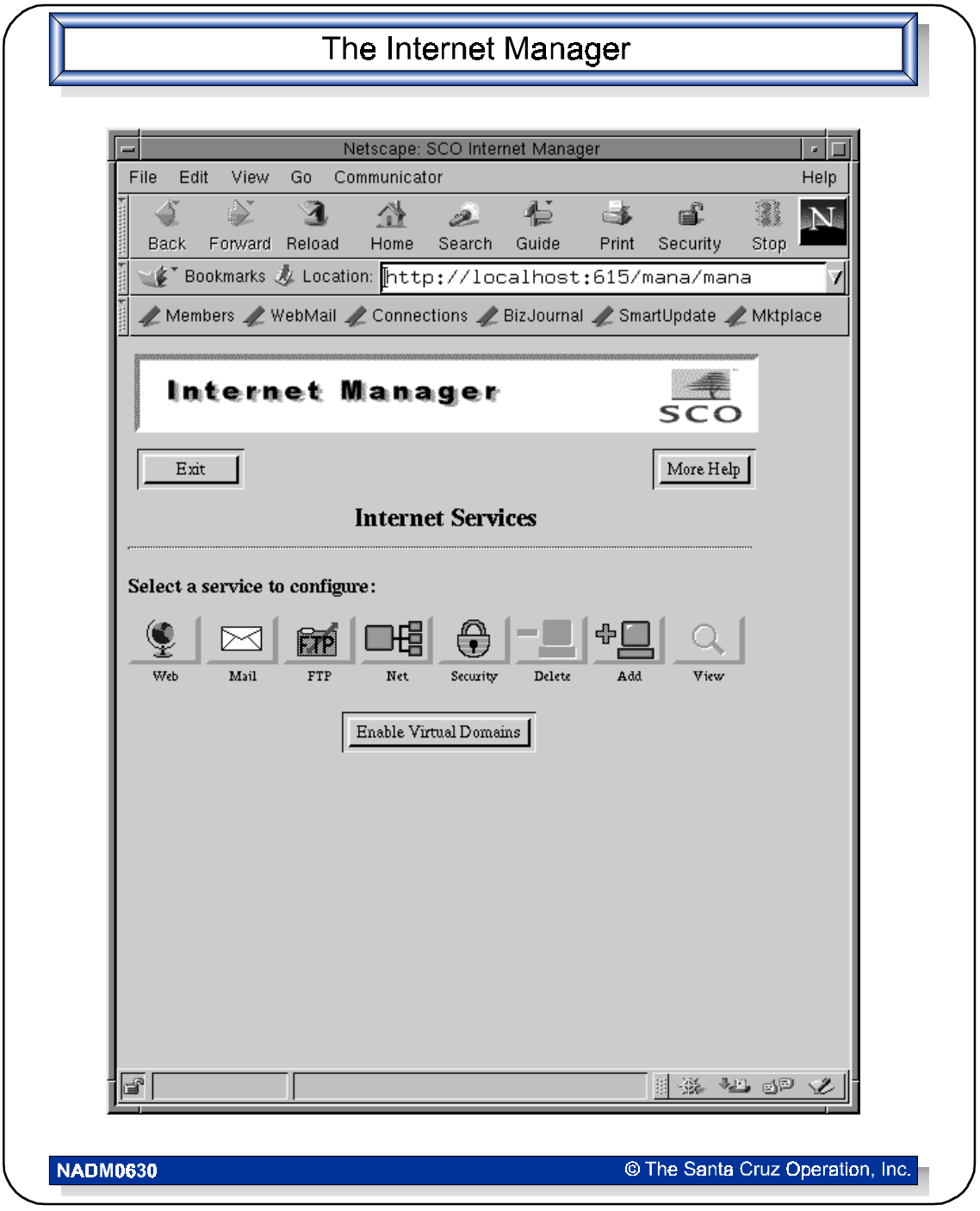
There are numerous configuration parameters that enhance the usability and security of SCO PPP. These are configured by:

- Selecting the **Advanced Options** button on each link configuration form used by the **Network Configuration Manager**
- Manually editing additional PPP configuration files, including:

/etc/pppauth	Specifies PAP and CHAP protocols configuration for connection authentication
/etc/pppfilters	Allows **pppd** to select which packets are transmitted or received over a PPP interface. See Section 10.2, “Packet filters,” on page 10-9 for more information.
/etc/ppppool	Provides a dynamic IP addressing scheme for an incoming SCO PPP connection

To learn more about using these options, see **scohelp ⇨ Networking Documentation ⇨ Networking Guide ⇨ Configuring the Point-to-Point Protocol (PPP)**.

AV 6-30


The Internet Manager
Netscape: SCO Internet Manager
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Exit
More Help
Internet Services
Select a service to configure:
Web Mail FTP Net Security Delete Add View
Enable Virtual Domains
NADM0630
© The Santa Cruz Operation, Inc.

6.5 Configuring PPP from Morning Star

To use PPP from Morning Star, you must first install the “PPP for Morning Star” software package from the SCO OpenServer Release 5 CD-ROM. It is not necessary to remove existing UUCP and PPP configuration files, if SCO PPP is configured. These can be used as guides for configuring Morning Star PPP.

You might want to demonstrate the processes of removing SCO PPP and installing PPP from Morning Star on the instructor machine. Time permitting, this could also be done as an exercise. However, there is an exercise for configuring packet filters in Lesson 10: “Network Security” that assumes SCO PPP is installed.

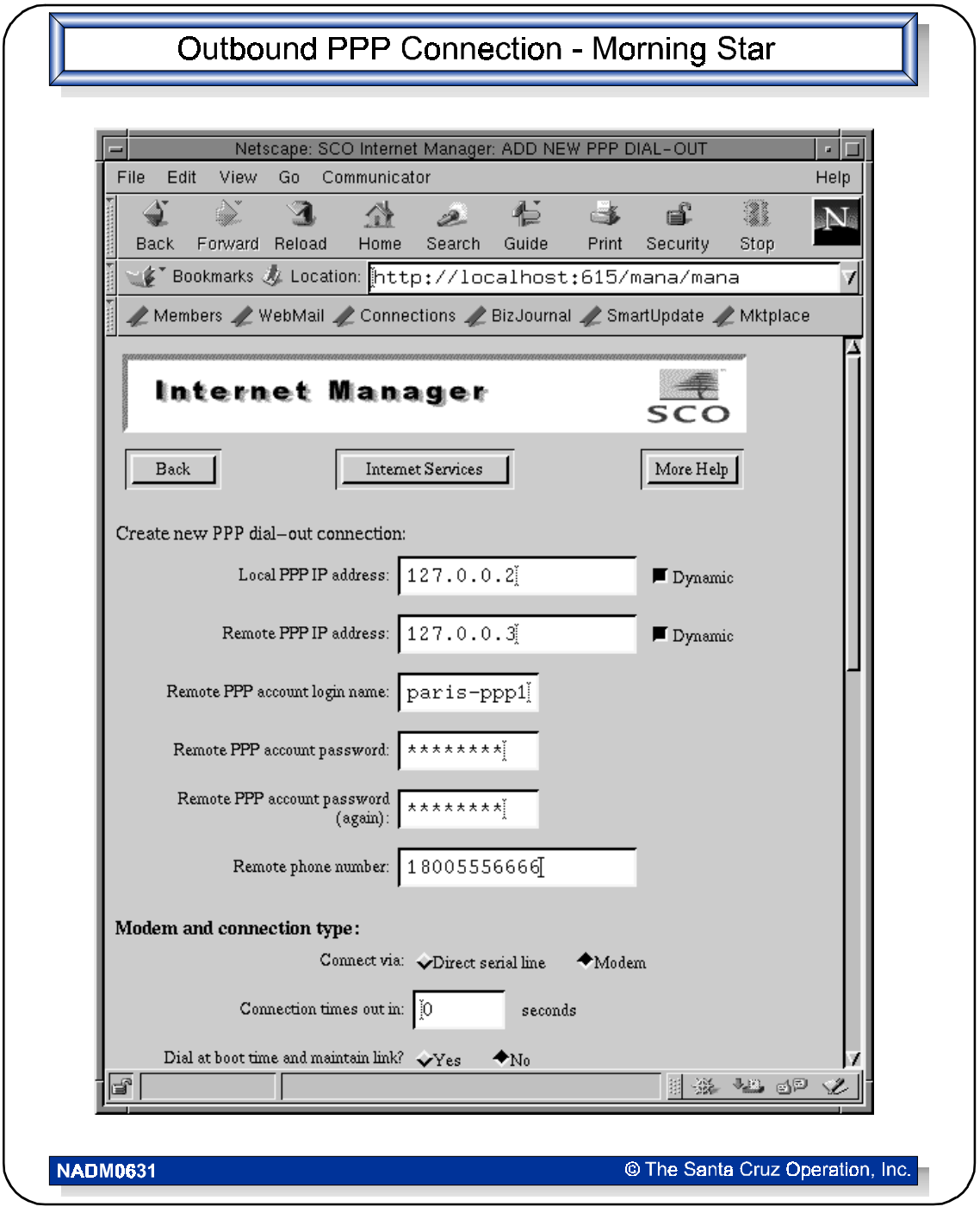
After installation, Morning Star PPP is configured using the **Internet Manager.** The **Internet Manager** performs these tasks:

- Creates dialer entries in */usr/lib/mstppp/Devices*
- Creates dial-out entries in */usr/lib/mstppp/Systems* for outgoing connections
- Adds entries to */usr/lib/mstppp/Autostart* for starting the **pppd** process
- Adds entries to */usr/lib/mstppp/Accounts* for incoming connections
- Adds entries to */usr/lib/mstppp/Auth* if authentication is specified
- Starts the PPP daemon, */usr/lib/mst/pppd* and brings up the configured outgoing interfaces

To run the **Internet Manager**, you must enter the administrative user ID and password. By default, the user name is **admin** and the password is the same as the **root** password.

AV 6-30: Internet Manager
This slide shows the **Internet Manager** before any connections are configured.

AV 6-31


Outbound PPP Connection - Morning Star
Netscape: SCO Internet Manager: ADD NEW PPP DIAL-OUT
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Back
Internet Services
More Help
Create new PPP dial-out connection:
Local PPP IP address: 127.0.0.2 Dynamic
Remote PPP IP address: 127.0.0.3 Dynamic
Remote PPP account login name: paris-ppp1
Remote PPP account password: ********
Remote PPP account password (again): ********
Remote phone number: 18005556666
Modem and connection type:
Connect via: Direct serial line Modem
Connection times out in: 0 seconds
Dial at boot time and maintain link? Yes No
NADM0631
© The Santa Cruz Operation, Inc.

6.5.1 Outbound Morning Star PPP connections

To create an outgoing Morning Star PPP connection:

Step 1. Start the **Internet Manager**. Use **admin** as the user ID and enter the **root** password.

Step 2. Click on **OK** to choose the "Add a PPP connection" selection, displayed by default when the **Internet Manager** is first started.

Otherwise, click on the **Net** icon and then **PPP Connections ⇨ Add New Dial-out PPP Connection**.

Step 3. Fill in the form, **Create new PPP dial-out connection**:

Local PPP IP address
: Enter an IP address for the local system. If the remote system assigns an IP address, toggle "Dynamic." The address is ignored (but must be entered).

Remote PPP IP address
: Enter an IP address for the remote system. Toggle "Dynamic" if applicable. The address is ignored (but must be entered).

Remote PPP account password
: Enter the password for your account on the remote system.

Remote phone number
: Enter the phone number of the remote system. Be sure to include dashes (-).

Step 4. Fill in the form **Modem and connection type** using these guidelines:

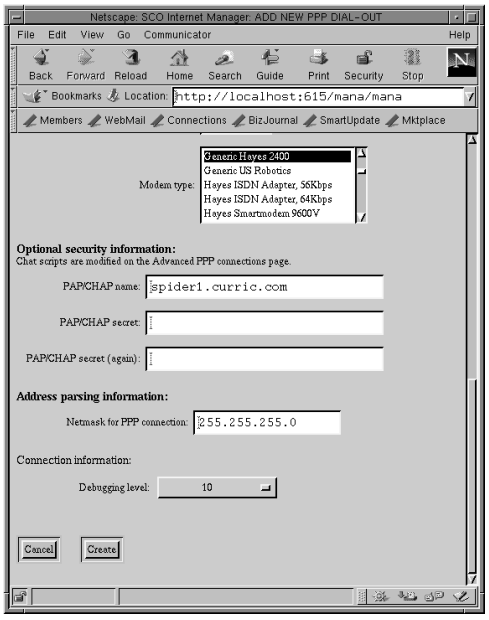
Connect via
: Toggle "Modem" unless you have a direct serial connection.

Connection times out in:
: Enter the number of seconds the connection needs to be idle before it hangs up.

AV 6-31: Outbound PPP Connection—Morning Star PPP
This slide shows the **Internet Manager** screen used to create a new PPP dial-out connection.

AV 6-32

Outbound PPP Connection - Morning Star (Continued)



NADM0632 © The Santa Cruz Operation, Inc.

AV 6-33

Outgoing PPP Connection - Morning Star Configuration Files

/usr/lib/mstppp/Autostart

```
/usr/lib/mstppp/dialout 127.0.0.2~:127.0.0.3~ auto exec
/usr/lib/mstppp/exec.dialout netmask 255.255.0.0 idle 60
```

/usr/lib/mstppp/Systems

```
hayes2400 tty1A  38400
```

/usr/lib/mstppp/Autostart

```
127.0.0.3 Any;5 ACU 38400  18005556666 "" "" ogin:--login:
\ppp-paris1 word: \q1Q8kyes!
```

NADM0633 © The Santa Cruz Operation, Inc.

Dial at boot time and maintain link?
Toggle "Yes' if this link is started at boot time.

Serial port speed — Select the baud rate for your port.

Serial device — Enter the modem control device for the port.

Modem type — Select your modem from the list. If not listed, try a generic entry such as `hayes2400`.

Step 5. Fill in the form, **Optional security information,** if applicable:

PAP/CHAP name — Enter the authentication name for this connection.

PAP/CHAP secret — Enter the authentication secret for this connection.

Step 6. Enter an appropriate netmask. This must match the netmask used on the remote server. If unknown, use the default 255.255.255.0.

Step 7. Set the debug level.

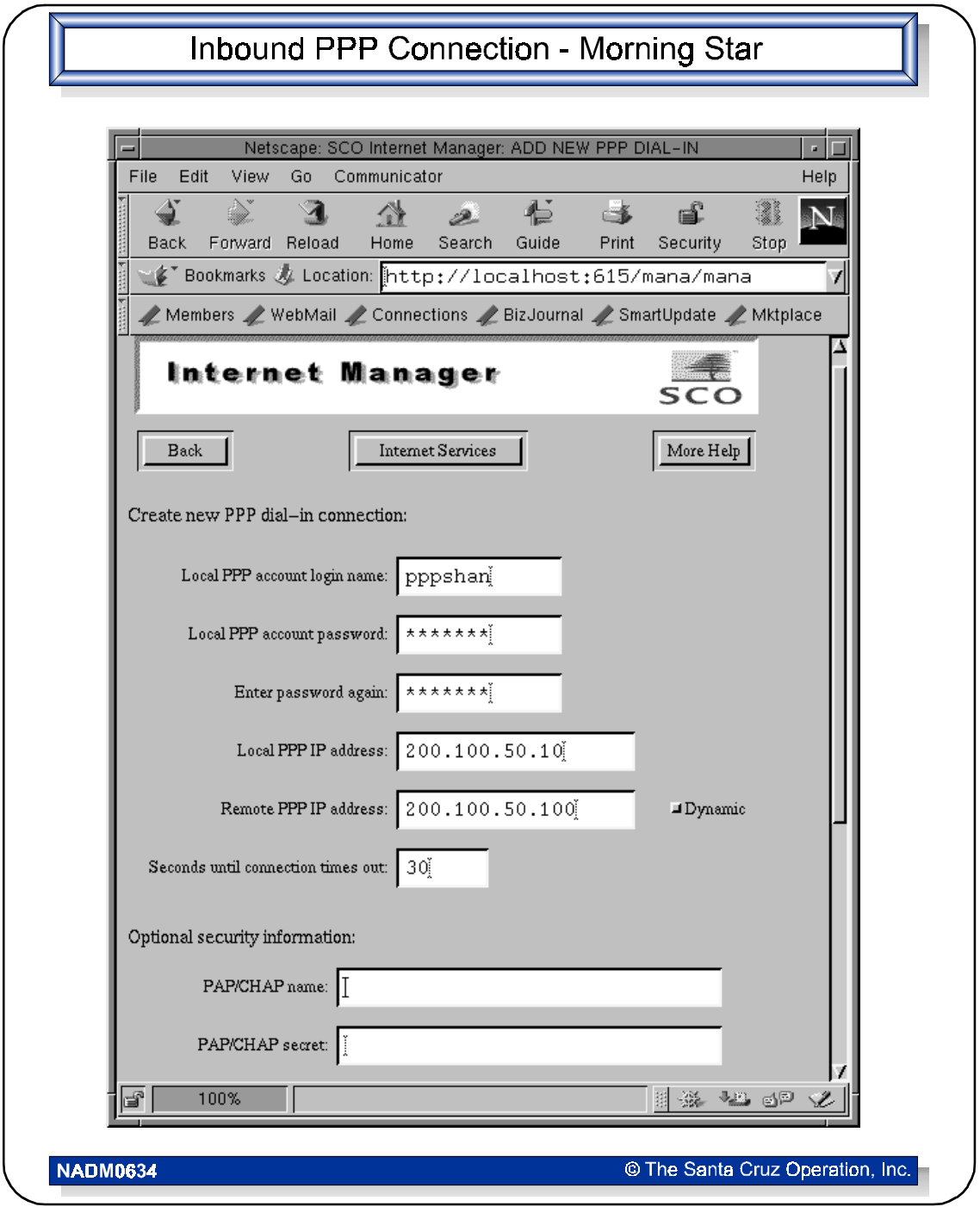
Step 8. Click on **Create** when finished.

The **Internet Manager** modifies the Morning Star configuration files as shown in AV 6-33. To test the connection, see Section 6.4.3.4, "Initiating a dynamic outgoing PPP connection," on page 6-55.

AV 6-32: Outbound PPP Connection—Morning Star PPP (Continued)
This slide shows the remaining fields needed for configuring outbound Morning Star PPP connections.

AV 6-33: Outbound PPP Connection—Morning Star Configuration Files
This slide shows the files modified by the **Internet Manager** for a dial-out connection.

AV 6-34


Inbound PPP Connection - Morning Star
Netscape: SCO Internet Manager: ADD NEW PPP DIAL-IN
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Back
Internet Services
More Help
Create new PPP dial-in connection:
Local PPP account login name: pppshan
Local PPP account password: *******
Enter password again: *******
Local PPP IP address: 200.100.50.10
Remote PPP IP address: 200.100.50.100
Dynamic
Seconds until connection times out: 30
Optional security information:
PAP/CHAP name:
PAP/CHAP secret:
100%
NADM0634
© The Santa Cruz Operation, Inc.

6.5.2 Inbound Morning Star PPP connections

To create an incoming Morning Star PPP connection:

Step 1. Start the **Internet Manager**. Use **admin** as the user ID and enter the **root** password.

Step 2. Click on **OK** to choose the "Add a PPP connection" selection, displayed by default when the **Internet Manager** is first started.

If this is not the first time the **Internet Manager** has been started, click on the **Net** icon and then **PPP Connections ⇨ Add New Dial-in PPP Connection**.

Step 3. Fill in the form, **Create new PPP dial-in connection** using these guidelines:

Remote PPP account login name

Enter the account name used by the remote system to login into the local system. This must be a new account name that does not already exist.

Remote PPP account password

Enter the password for your account on the remote system.

Local PPP IP address

Enter an IP address for the local system. This must be a unique address that is not being used for any other networking interface.

Remote PPP IP address

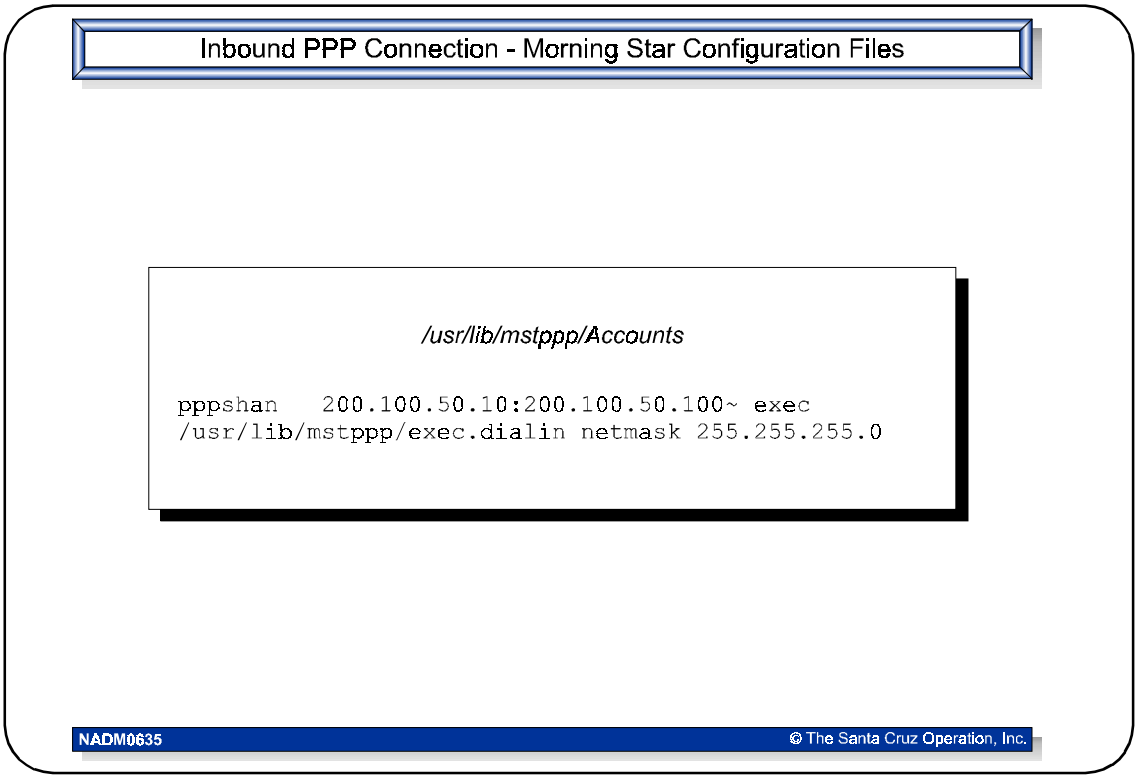
Enter an IP address for the remote system. If the remote system assigns an IP address, toggle "Dynamic." The address is ignored (but must be entered).

Seconds until connection times out

Enter the number of seconds the connection needs to be idle before it hangs up. Enter '0' (zero) to keep the link connected continuously.

AV 6-34: Inbound PPP Connection—Morning Star PPP
This slide shows the remaining fields needed for configuring outbound Morning Star PPP connections.

AV 6-35


Inbound PPP Connection - Morning Star Configuration Files
/usr/lib/mstppp/Accounts
pppshan 200.100.50.10:200.100.50.100~ exec
/usr/lib/mstppp/exec.dialin netmask 255.255.255.0
NADM0635
© The Santa Cruz Operation, Inc.

Step 4. Fill in the form **Optional security information** if applicable:

PAP/CHAP name — Enter the authentication name for this connection.

PAP/CHAP secret — Enter the authentication secret for this connection.

Step 5. Enter an appropriate netmask. This must match the netmask used on the remote server. If unknown, use the default 255.255.255.0.

Step 6. Set the debug level.

Step 7. Click on **Create** when finished.

The **Internet Manager** modifies the Morning Star configuration file */usr/lib/mstppp/Accounts*, as shown in AV 6-35. Test the connection by dialing in from another system.

AV 6-35: Inbound PPP Connection—Morning Star PPP
This slide shows the remaining fields needed for configuring outbound Morning Star PPP connections.

6.5.3 Advanced Morning Star PPP configuration

The are numerous configuration parameters that enhance the usability and security of PPP from Morning Star. These are configured by editing these files:

/usr/lib/mstppp/Filter	Allows **pppd** to select which packets should be transmitted or received over a PPP interface
/usr/lib/mstppp/IPPool	Provides a dynamic IP addressing scheme for an incoming SCO PPP connection
/usr/lib/mstppp/Login	PPP login description file format

To learn more about using these options, see **scohelp ⇨ SCO Internet Family Documentation ⇨ SCO PPP from Morning Star User's Guide**.

6.6 Troubleshooting PPP

Configuring systems to communicate can be a complicated task. A problem that occurs at any step of the configuration process can cause a PPP session to fail. Problems with PPP negotiations, especially on new configurations is not rare.

It is important to diagnose which component or parameter is failing in order 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.

6.6.1 Testing the connection with cu(C)

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:

- Call 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.
- Call the system using **cu**(C) 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.
- Use the **cu** command to place an outgoing call.

To test the connection between the systems, try the following 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*(SFF) file are probably configured correctly.

- **cu *sysname***

  This command again tries to contact the remote machine but this time refers to the *Systems*(SFF) 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**(C) does not use the login script specified in the *Systems*(SFF) 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.

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, add a “Direct” entry to the */usr/lib/uucp/Devices* file. For example:

```
Direct tty1a - 38400 direct
```

The `Direct` entry must follow the `ACU` entry.

To connect to a modem on your first serial port, enter:

```
cu -l tty1a dir
```

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

For more on troubleshooting modem connections, see **scohelp ⇨ SCO OpenServer Handbook ⇨ Troubleshooting modems**.

6.6.1.1 Testing a login script

By default, **cu** does not use the login script specified in the *Systems*(F) file. Run **uutry** to connect using the login script:

```
/usr/lib/uucp/uutry -x9 -r sysname
```

The **-x9** option enables full debugging writing to a log file named */tmp/sysname*. If the previous commands were successful and this one fails, check the login script.

6.6.2 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.

There are three debugging levels that can be configured using the **Advanced options** button in the **Network Configuration Manager** PPP configuration screen:

`0` No *info* messages are logged.

`1` Logs LCP, IPCP, PAP, and CHAP negotiation information

`2` Logs all negotiation information

The debug parameter can also be set manually:

1. Modify the entry in */etc/ppphosts* for the connection to add a `debug` entry:

```
127.1.1.2:127.1.1.3 attach=isp uucp=isp debug=2
```

2. Signal the PPP daemon to read the modified */etc/ppphosts* file:

```
kill -1 `cat /etc/pppd.pid`
```

After logging is configured, re-establish the PPP connection and view the PPP information logged in */usr/adm/syslog*.

Logging of debugging information can also be enabled while **pppd** is running using the command line:

```
kill -16 ‘cat /etc/pppd.pid‘
```

To stop logging, enter:

```
kill -17 ‘cat /etc/pppd.pid‘
```

For more information about the debug levels, see **pppd**(ADM). Help interpreting the PPP log file can be found in **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Point-to-Point Protocol (PPP) ⇨ Troubleshooting PPP negotiation.**

Morning Star PPP logs information to */usr/adm/pppd.log*. See *PPPD*(MST_PPP).

6.6.3 Checking the PPP interface

If the PPP log file shows that the PPP link has been established, you can then check that the correct network interface has been added to the system:

Step 1. Use the **ifconfig**(ADMN) command with the **-a** option to check if an interface (such as `ppp0`) for the link exists.

Step 2. Verify the displayed flags indicate that it is marked UP and 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.

6.6.4 Verifying network connections

If the interface appears to be up, verify network connectivity between the local and remote host. You can do this with **ping**(ADMN). It is tailor-made for this purpose:

```
ping -n remote_IP_address
```

The **-n** option specifies numeric mode only. The command will not try to look up names that go along with IP addresses in its output. This can be necessary if the DNS server is on the remote side.

- 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 Documentation Set ⇨ Configuring Network Connections ⇨ Troubleshooting network configuration**.
- If **ping** succeeds, you have a working connection between the tested machines. You should test several systems, including some across routers.

6.6.5 Checking 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 5-7.
- The remote machine is not doing proxy ARP. Check the configuration of this interface on the remote machine. Running **arp** with the **-a** flag 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**(ADMN) command. For example, to set up a default route:

  ```
  route add default remote_IP_address
  ```

For more information, refer to **scohelp ⇨ Networking Documentation Set⇨ Networking Guide⇨ Configuring Internet Protocol (IP) routing⇨ Troubleshooting IP routing.**

6.6.6 Reseting PPP

If all else fails, try resetting a link or even restarting the PPP daemon.

To bring a PPP connection down, use **ifconfig**(ADMN). For example:

ifconfig ppp0 down

To stop the ppp daemon, enter:

```
kill -9 ‘cat /etc/pppd.pid‘
```

To restart the ppp daemon, enter:

```
/etc/pppd
```

6.7 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 using both SCO PPP and PPP from Morning Star, was 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 SCO PPP WAN connections? Which ones specify the phone numbers and login scripts used to contact remote systems.

3. Which file contains a list of all SCO PPP link and its associated parameters?

4. Which file contains parameters for SCO PPP PAP and CHAP authentication?

5. Which file contains parameters for assigning IP addresses to incoming SCO connections?

6.8 Lab: Configuring PPP Connections

The goal of this lab is to configure incoming and outgoing PPP connections using the **Network Configuration Manager**.

Two computers are needed for this lab, one for the outgoing connection and one for the incoming connection.

This lab is probably best done with pairs of students working together.

Preparation

Before beginning this lab, your instructor will supply you with the following pieces of 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.

Instructor Guide

6.8.1 Disabling LAN access

Before configuring an outgoing PPP connection, 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:

```
ifconfig net0 down
route flush
```

After the lab, reboot your system to re-enable your network interface and re-initialize your routing table.

6.8.2 Configuring a modem for outgoing calls

If you have not already done so, configure a modem on the system that is used for the outgoing connection.

Procedure

Step 1. Start SCOadmin.

Step 2. Select **Hardware** ⇨ **Modem Manager**.

Step 3. Select **Modem** ⇨ **Add** ⇨ **Manual Configuration**.

Step 4. Select your “Modem vendor” and “Modem model” from the lists.

Step 5. Select the port the modem is using in the “Modem port” field.

Step 6. Click on **Configure port**.

Step 7. Select **outgoing only**.

Step 8. Select a “Speed” of 57600 bps.

Due to a bug in the **Modem Manager**, the speed selection might not be recorded in the */etc/uucp/Devices* file. You might need to verify the speed setting in this file. On some systems, the modem will not function unless set to a speed of 57600 or lower.

Step 9. Click on **OK** twice.

Step 10. Select **Host** ⇨ **Exit**.

Step 11. 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 ENTER to see the login prompt.

Step 12. View the new entry in */usr/lib/uucp/Devices*. This is a reference for creating an entry in */usr/lib/uucp/Systems*.

6.8.3 Configuring */usr/lib/uucp/Systems*

Modify the */usr/lib/uucp/Systems* file to add an entry with the phone number of the remote system.

Procedure

Step 1. Edit the */usr/lib/uucp/Systems* file using **vi**(C) or **scoedit**(XC).

Step 2. Add a line with these fields:

Site name	Enter a name for the remote site. This does not have to be the actual remote system name.
Schedule	Enter "Any" to allow calls at any time.
Device type	Enter "ACU" as the type.
Speed	Enter the speed used in */usr/lib/uucp/Devices*.
Phone	Enter the phone number for the remote system
Login script	Enter a login script using the login name and password provided by your instructor. An example entry for a remote UNIX system is:

```
ogin: nppp ssword: nppp-passw
```

Step 3. When finished, your entry should be similar to this:

```
sitename Any ACU 38400 555-5555 ogin:nppp ssword:nppp-passw
```

Step 4. Exit and save changes to the file.

Step 5. Test your connection with the command:

cu *sitename*

If you see a login prompt, the test was successful. You might have to press ENTER to see the login prompt.

Step 6. Verify that your login script works. Enter:

/usr/lib/uucp/uutry -x9 *sitename*

6.8.4 Configuring an outgoing PPP connection

Configure your system to call another machine to initiate a PPP connection using the **Network Configuration Manager**.

In this part of the lab, configure your system to call a system previously configured by your instructor. This allows you to test your outgoing PPP connection by itself. Later in the lab you will configure an incoming system as well.

Procedure

To configure a manual outgoing PPP connection:

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

Step 2. Select **Hardware ⇨ Add new WAN connection.**

Step 3. Select "SCO TCP/IP PPP driver" from this list of WAN connection types and click on **Continue**.

Step 4. Select "SCO TCP/IP" as the networking product to be added to the PPP driver and click on **Add**.

Step 5. Choose "Manual Outgoing" as the connection type and click on **OK**.

Step 6. Fill in the configuration form:

Attach name	This is the name that is used to bring up the connection with the **pppattach** command.
Local host name	Enter a name for the local PPP connection. Make sure this name is not used by another network interface.
Host IP address	Enter the IP address provided by the instructor or 0.0.0.0 if the remote system is providing IP addresses.
Destination name	Enter a name for the local PPP connection. Make sure this name has not been used.
Destination IP address	Enter the IP address provided by the instructor or 0.0.0.0 if the remote system is providing IP addresses.
Netmask	Enter the netmask information provided by the instructor or leave blank if IP addresses are assigned remotely.
UUCP destination name	Enter the site name of the remote systemconfigured in the */usr/lib/uucp/Systems* file.

Number of UUCP retires
: Specify the number of times UUCP tries to connect to the remote system.

Gateway:
: Select "No."

Step 7. Click on **OK** when the form is filled in.

Step 8. Click on **No** when asked to configure more links.

Step 9. Click on **OK** when the successful configuration message displays.

Step 10. Select **Hardware ⇨ Exit.**

Step 11. Click on **Yes** to relink the Unix kernel when prompted. Answer "y" to the prompts to boot the new kernel by default and rebuild the kernel environment.

Step 12. Press ENTER to continue when the kernel has been rebuilt, and exit the **Network Configuration Manager**.

Step 13. Reboot the system.

Step 14. Initiate your connection with the command:

```
pppattach sitename
```

where *sitename* is the name you entered in step 6.

Step 15. Test your connection by using **ping**, **telnet**, and other networking commands to access systems on the remote side of the link.

6.8.5 Configuring a modem for incoming calls

If you have not already done so, configure a modem on the system that is used for the incoming connection. Follow the procedure in Section 6.8.2 on page 6-88, this time allowing incoming calls in Step 7.

If you have already configured the modem, use the **Modem Manager** to modify the configuration to allow incoming calls.

Procedure

Step 1. Start the **Modem Manager** as **root**.

Step 2. Select **Modem** ⇨ **Modify**.

Step 3. Select “On” for Answer Mode.

Step 4. Click on **OK**.

Step 5. Select **Host** ⇨ **Exit**.

6.8.6 Configuring an incoming PPP connection

Using the **Network Configuration Manager**, configure a system to answer incoming PPP calls. Do not use the same system you used in the outgoing connection lab.

Procedure

Step 1. If you do not already have the PPP driver configured, follow the procedure in Section 6.8.4 on page 6-90, noting these differences:

- In Step 4, select “Dynamic incoming” as the PPP connection type.
- In Step 6, note the differences in the link configuration parameters when configuring a Dynamic link. Follow the guidelines for Step 8 below.

Step 2. If PPP is already configured, you need only modify the existing protocol to add a new endpoint for the incoming connection:

Step 3. Start the **Network Configuration Manager** as **root**.

Step 4. Select “SCO TCP/IP” under “SCO TCP/IP PPP driver.”

Step 5. Select **Protocol** ⇨ **Modify protocol configuration.**

Step 6. When the currently configured links are displayed, toggle “Add” and click on **OK**.

Step 7. Toggle “Dynamic Incoming” and click on **OK**.

Step 8. Fill in the form shown using these guidelines:

PPP login name:	Enter the user name **ppptest**.
Local host name	Enter a name for this incoming link. This name must be different from any other networking interface configured on this system.
Host IP address:	Enter an unused IP address for the local IP address.
Destination name	Enter a name for the remote system. The choice of the name is arbitrary (but must be unique on this system).
Host IP address:	Enter an unused IP address for the remote IP address.

Step 9. Click on **OK** when finished.

Step 10. The user account is now created using the account name provided in the configuration form.

When prompted, type "n" to the prompt for specifying a user ID. Press ENTER to continue.

Step 11. Set the password for the new PPP login account. to `Safe1!`.

Step 12. Click on **No** when asked to configure more links.

Step 13. Click on **OK** to confirm the successful configuration.

Step 14. Select **Hardware ⇨ Exit.** You are not prompted to relink the kernel if the PPP driver is already configured.

When finished, a new entry is added to the */etc/ppphosts* file for the dynamic incoming link and a new user account is created with the login and password you specified, and a login shell of */usr/lib/ppp/ppp*.

6.9 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

cu(C)	Connects to a configured device or system.
netconfig(ADM)	Network configuration utility which can be invoked from the command line, or via **scoadmin**(ADM)
ping(ADMN)	Tests network connections by eliciting hosts to reply to packets sent to them
pppattach(ADMN)	Brings up *manual* or *on demand* outgoing PPP connections
pppd(ADMN)	Is the PPP daemon
route(ADMN)	Allows manually manipulation of the routing tables
uutry(ADM)	Connects to a configured device or system using the login script specified in the */usr/lib/uucp/Systems* file

Relevant files

/usr/lib/uucp/Devices	Contains one entry per device to be used by UUCP. Refer to the *Devices*(F) manual page.
/usr/lib//uucp/Systems	Contains one entry per system connected via UUCP. Refer to the *Systems*(F) manual page.
/etc/ppphosts	Contains one entry per SCO PPP connection (or endpoint). Refer to the *ppphosts*(SFF) manual page.
/etc/pppauth	Is the PPP PAP/CHAP authorization file for SCO PPP. Refer to the *pppauth*(SFF) manual page.
/etc/pppfilter	Is the PPP packet filter specification file. Refer to the *pppfilter*(SFF) manual page.

/etc/ppppool	Contains address pools for assigning IP addresses to remote connections. Refer to the *ppppool*(SFF) manual page.
/etc/pppd.pid	Contains the process ID (PID) of the PPP daemon, **pppd**.
/etc/hosts	Contains one entry per PPP connection (or endpoint). Refer to the *hosts*(F) manual page.
/etc/passwd	User account database containing PPP login account. Refer to the *passwd*(F) manual page.
/etc/tcp	Contains PPP startup script. Refer to the **tcp**(ADMN) manual page.
/etc/inittab	Contains initialization data for the serial ports. Refer to the *inittab*(ADMN) manual page.
/usr/lib/mstppp/Accounts	Lists inbound PPP accounts
/usr/lib/mstppp/Auth	Is the PPP PAP/CHAP authorization file. Refer to the *ppp.AUTH*(MST_PPP) manual page.
/usr/lib/mstppp/Autostart	Is the startup file for initiating outbound PPP daemons
/usr/lib/mstppp/Devices	Is the PPP outbound physical device description file format
/usr/lib/mstppp/Dialers	Is the PPP dialer description file format. Refer to the *ppp.DIALERS*(MST_PPP) manual page.
/usr/lib/mstppp/Filter	Is the PPP packet filter specification file format. Refer to the *ppp.FILTER*(MST_PPP) manual page.
/usr/lib/mstppp/Login	Is the PPP login description file format.
/usr/lib/mstppp/Systems	Is the PPP neighboring systems description file format. Refer to the *ppp.SYSTEMS*(MST_PPP) manual page.
/usr/lib/mstppp/IPPool	Is the PPP neighboring systems description file format
/etc/rc2.d/S89mstppp	Is the PPP startup script performed for multiuser environment

6.10 Answers to Review Questions

This section provides suggested 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 to. 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. The */etc/ppphosts* file contains parameters for SCO PPP authentication using PAP and CHAP.

4. The */etc/pppauth* file contains parameters for SCO PPP authentication using PAP and CHAP.

5. The */etc/ppppool* file contains parameters for assigning IP addresses to PPP connections.

7

TCP/IP Administration Topics

Instructor Guide

Lesson 7

TCP/IP Administration Topics

Suggested lesson duration

This lesson should take approximately 3 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

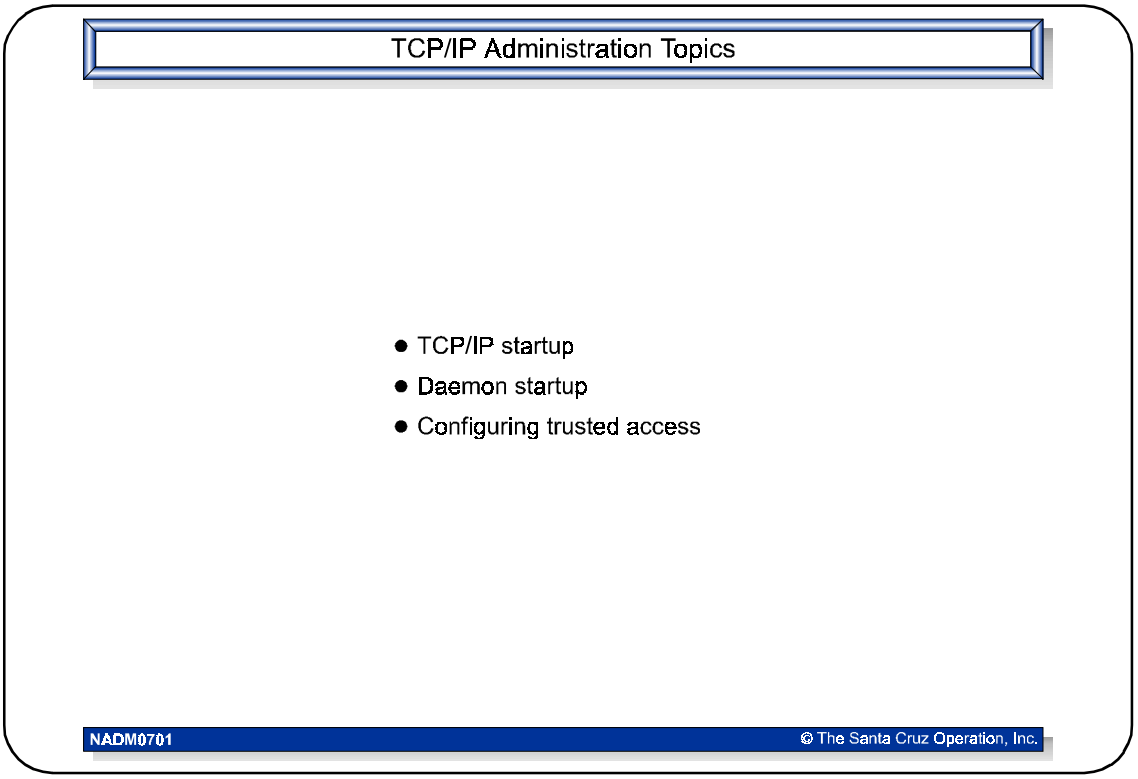
Your classroom must have a working network for this lesson.

Set up a **guest** account on each host to perform trusted access exercises.

Instructor preparation

In many sections in this lesson, files are referred to that might have *netgroup* entries if NIS is activated. If students are aware of *netgroups* and NIS and they are asking questions, refer them to **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Chapter 9: Configuring the Network Information Service (NIS).** You should, of course, be aware of which files can accept netgroup notation.

AV 7-1


TCP/IP Administration Topics
TCP/IP startup
Daemon startup
Configuring trusted access
NADM0701
© The Santa Cruz Operation, Inc.

Introduction

Previous lessons have discussed how to design and build a LAN and WAN using the SCO OpenServer™ product. 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 concepts and can accomplish the tasks presented in this chapter, you will be able to perform your job as a network administrator with greater ease.

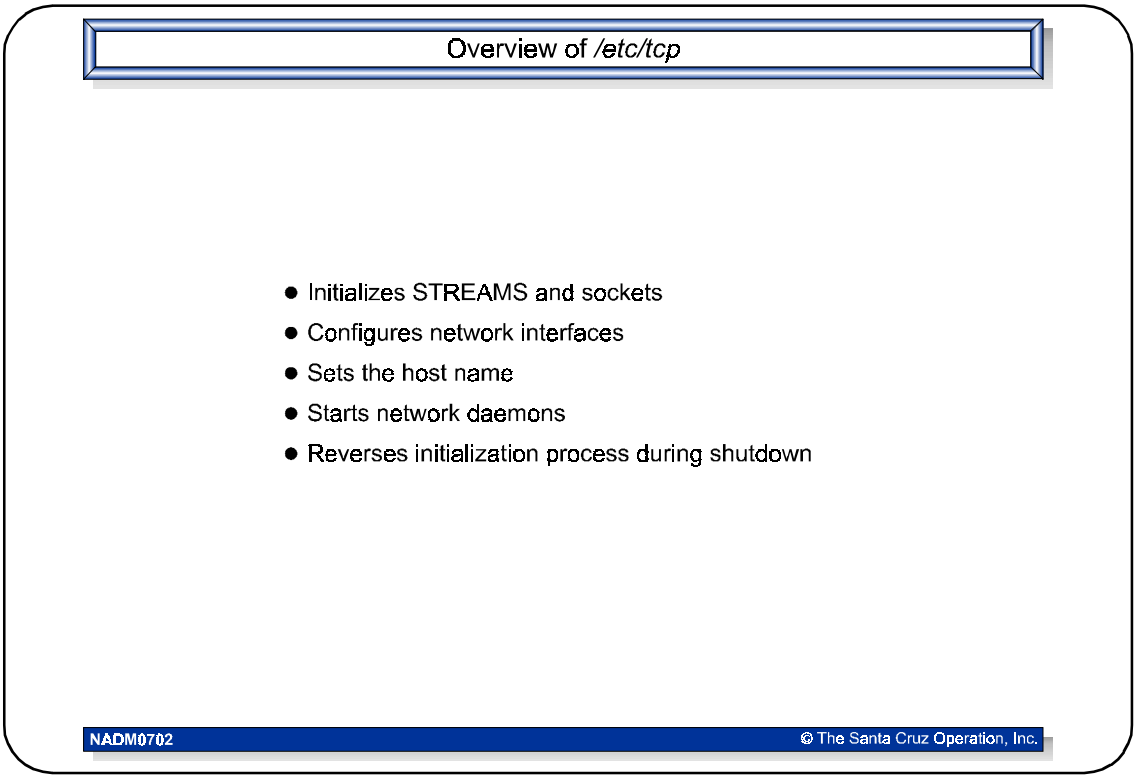
Objectives

This lesson covers in detail general TCP/IP administration issues. By the end of this lesson, you will be able to:

- Describe the role of the */etc/tcp* file in initializing the TCP/IP environment
- Explain what the **inetd** daemon does and how to add, delete, and reconfigure services started by **inetd**
- Use **telnet** to test daemon functionality
- Configure and explain the functionality of trusted user access

AV 7-1: TCP/IP Administration Topics
The first slide describes the overall topics covered in this lesson.

AV 7-2


Overview of /etc/tcp
Initializes STREAMS and sockets
Configures network interfaces
Sets the host name
Starts network daemons
Reverses initialization process during shutdown
NADM0702
© The Santa Cruz Operation, Inc.

7.1 TCP/IP Start-up

TCP/IP is started when your system enters multiuser mode and stopped when you shut down to single-user mode by. Understanding the TCP/IP startup and shutdown process will help you troubleshoot problems that you might encounter.

7.1.1 The */etc/tcp* file

AV 7-2: Overview *of /etc/tcp*
Use this slide as an introduction to the sections that follow.

The */etc/tcp* file is a shell script that enables and disables TCP/IP, including:

- Configuring the necessary STREAMS drivers needed to support TCP/IP
- Initializing the network interfaces with **ifconfig**(ADMN)
- Setting the host name
- Starting and stopping the TCP/IP-related daemons

To start and stop TCP/IP, as **root**, run the */etc/tcp* script from the command line. To do this, enter:

```
tcp [ start | stop ]
```

There are symbolic links to */etc/tcp* from files in both the */etc/rc2.d* and */etc/rc0.d* directories. These links cause */etc/tcp* to be run both with the **start** or **stop** options, as the system goes in and out of multiuser mode.

NOTE: The link is from the *rc* files to the */etc/tcp* file, not the other way around. Also, note that the filename in the *rc* directories may be version-dependent. For example, SCO® Open Desktop® 3.0 linked */etc/rc0.d/K85tcp* and OpenServer Release 5 links */etc/rc0.d/K96tcp*. Because of SSOs in OpenServer Release 5, the links are not actually to */etc/tcp*, but to somewhere under */var/opt/K/SCO*.

Have the students find the links to */etc/tcp* in the *rc0.d* and *rc2.d* directories.

Whenever network interfaces are added or removed using the **Network Configuration Manager**, the */etc/tcp* file is modified by adding or removing **ifconfig** commands in the script. These same actions also modify the */etc/strcf* file.

ifconfig and */etc/strcf* will be discussed later.

AV 7-2 illustrates the basic process of starting TCP/IP. The following sections show the details of how the */etc/tcp* script starts and stops TCP/IP.

7.1.2 Functions performed by */etc/tcp*

The configuration of the TCP/IP kernel support is done only once during system boot. This support involves setting up a stack of STREAMS modules needed by TCP/IP.

Other things that the */etc/tcp* script does only once include setting the host name and configuring the network interfaces.

7.1.2.1 Setting the host name

After STREAMS and sockets are initialized, the */etc/tcp* script sets the host name.

The host name is derived by taking the node name of the system (obtained with the command line `uname -n`), and concatenating the domain name, stored in */etc/default/tcp*.

Both of these values are initially set when you install the system. Changing the domain and host names is not as simple as running the **uname** command or editing the TCP configuration files, */etc/default/tcp* and */etc/tcp*. Several subsystems use host and domain names, including:

- Mail
- Name resolution (DNS and */etc/hosts)*
- NIS
- Web servers

Because the consequences of changing the node or domain names are far-reaching, it is important to set them properly during installation. This exercise shows all the files that contain the host name.

Exercise: Locating all references to the host name

Step 1. Log in as **root.**

Step 2. Enter this command:

```
find / -type f -exec \
fgrep -l 'hostname' {} \; > /tmp/hostname
```

where ***hostname*** is the host name of your machine.

It is a good idea to run this as a background job.

Step 3. View the */tmp/hostname* file when the command is finished and discuss the results with your instructor.

AV 7-3

Example **ifconfig**(ADMN) 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
net1: flags=4043<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 200.100.50.2 netmask ffffff00 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
ppp0: flags=4071<UP,POINTOPOINT,WANTIOCTLS,RUNNING,MULTICAST> mtu 1500
        inet 200.100.50.3 --> 200.100.50.200 netmask ffffff00
        perf. params: recv size: 4096; send size: 8192; full-size frames: 1
```

Bringing interfaces down and back up:

```
# /etc/ifconfig ppp0 down
# /etc/ifconfig ppp0 up
```

NADM0703 © The Santa Cruz Operation, Inc.

7.1.2.2 Configuring networking interfaces

AV 7-3: Example ifconfig(ADMN) Commands
Use this slide to show students typical usage of **ifconfig** for configuring interfaces and bringing them up and down.

After the STREAMS, sockets, and host name are set, */etc/tcp* initializes each of your machine's network interfaces using the **ifconfig**(ADMN) command.

The **ifconfig** command is used to configure a network interface initially. It can also be used to change a network configuration, including temporarily taking the interface out of service. AV 7-3 shows the syntax of the **ifconfig** command.

NOTE: In addition to one **ifconfig** command per interface you previously configured, there is also an **ifconfig** command for the *loopback* interface.

The most common parameters set when initializing an interface are:

netmask Sets the netmask for this interface. If a machine is a router, (it is on two or more networks), it is possible that the netmasks may vary between interfaces.

broadcast Sets the broadcast address for this interface. If a machine is a router, each broadcast address is different.

Use the LCD or whiteboard to display one or more of the **ifconfig** commands from your */etc/tcp* file, and walk through each argument with the students. NOTE: The **-trailers** parameter is no longer used, and the **perf** parameter is new. Point out that the **destaddr** parameter is only used with SLIP and PPP connections, if the interface is not configured for pooling addresses. NOTE: Manual outgoing PPP connections do not have **ifconfig** entries in the */etc/tcp* file.

The *if.ignore* file no longer exists in OpenServer 5.

The */etc/tcp* script also hard-wires a multicast route for each non-loopback interface.

You can run **ifconfig -a** to list currently configured interfaces and their parameters.

7.1.3 Maintenance-mode */etc/tcp* activities

SCO TCP/IP allows you to perform simple administrative tasks using network utilities while in system maintenance mode. The previous section detailed the tasks performed by the */etc/tcp* script regardless of the system boot state.

When the system enters system maintenance mode, */etc/tcp* performs these additional actions that allow TCP/IP to interact with other systems with the same network number:

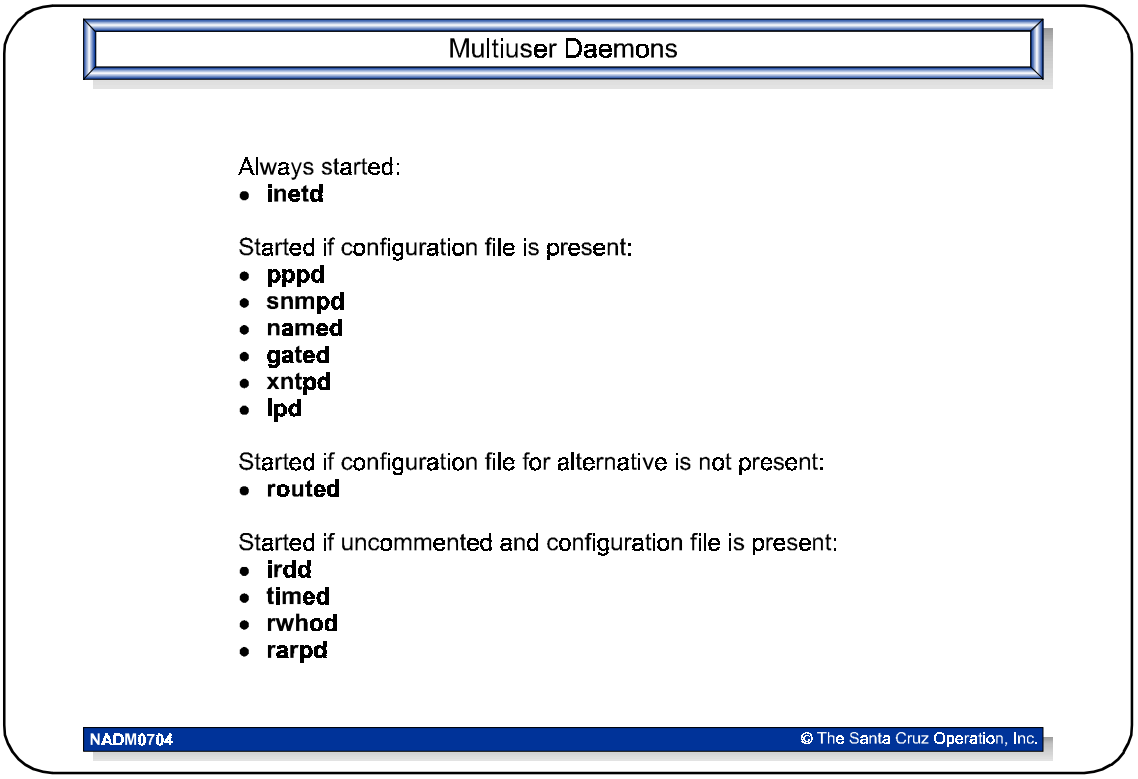
- Runs the **maskreply**(ADMN) command, which alerts other systems to your system netmask

- Starts the **strerr**(ADM) daemon, which logs error messages in files in the directory */usr/adm/streams*

- Starts or restarts the **syslogd**(ADM) daemon, which logs messages to a log file. This daemon is highly configurable and can be very helpful when you are trying to diagnose networking problems. The default configuration file for **syslogd** is the */etc/syslog.conf* file.

Have students look up the manual pages for **strerr**(ADM) and **syslogd**(ADM) for more information about error and message logging.

AV 7-4


Multiuser Daemons
Always started:
inetd
Started if configuration file is present:
pppd
snmpd
named
gated
xntpd
lpd
Started if configuration file for alternative is not present:
routed
Started if uncommented and configuration file is present:
irdd
timed
rwhod
rarpd
NADM0704
© The Santa Cruz Operation, Inc.

7.1.4 Multiuser-mode */etc/tcp* activities

AV 7-4: Multiuser Daemons
Use this slide to provide an overview of the TCP/IP daemons started to support multiuser operations. Point out that for many of these daemons, */etc/tcp* can be modified to have the daemon expect a different configuration file name. Refer students to the manual pages for daemons of interest.

In addition to the configuration performed and daemons started for TCP/IP in system maintenance mode, several other daemons are started to add functionality for multiuser mode. These daemons include:

inetd(ADMN) — The Internet super daemon (**inetd**) listens for connection requests with the network services daemons listed in */etc/inetd.conf* and starts those daemons on demand, thus saving memory and process table resources. More on **inetd** is presented in Section 7.2 on page 7-19.

pppd(ADMN) — The Point-to-Point Protocol daemon (**pppd**) only starts if you have configured an */etc/ppphosts* file. For more information about this daemon and file, see Lesson 6: "Configuring WAN Connections."

snmpd(ADMN) — The Simple Network Management Protocol (SNMP) daemon is started only if you have configured the */etc/snmpd.conf* file. Further discussion of SNMP is beyond the scope of this course.

named(ADMN) — The **named** daemon is only started if you have configured an */etc/named.boot* file. For further information about **named**, see Lesson 8: "Domain Name System (DNS) Configuration."

routed(ADMN) or **gated**(ADMN) — The **gated** daemon starts if */etc/gated.conf* is present. Otherwise **routed** starts by default. For more information on routing, see Lesson 5: "Routing."

xntpd(ADMN) — The Internet time daemon starts only if you have created an */etc/ntp.conf* file. This daemon is used to synchronize the kernel's clock using time servers on the Internet.

lpd(ADMN) — This remote line-printer daemon only starts if you have configured any of your printers as either print servers or clients, and thus have an */etc/printcap* file on your system. Configuring remote line printing is covered in the SCO System Administration courses.

7.1.5 Network daemons not started

Several daemons are not started by the */etc/tcp* script because their startup code is commented out. If you want any of these daemons to start, you must edit */etc/tcp* and remove the comment character (#) from their startup code. Before you do so, read the comments preceding each daemon's startup code. The following is a list of daemons that are not started by default:

irdd(ADMN) — This daemon is an alternate to either **routed** or **gated** and uses the Internet router discovery protocol. See the **irdd** manual page for more information.

timed(ADMN) — This daemon is an older and less reliable alternate to the **xntpd** daemon.

Students might want to know what the differences are between **xntpd** and **timed**. The short answer is, **xntpd** is more accurate, but takes more computing and networking resources. The long answer is that they need to read the respective manual pages and RFCs.

Instructor Guide

rwhod(ADMN) — This daemon broadcasts the information you see with the **w**(C) command using broadcast messages. This information can then be viewed from other machines using the **ruptime**(TC) command. Running this command on your machines can decrease network performance due to traffic and increased security risks.

The **rwhod** command may present an increased security risk for several reasons. First, it allows other people to find out valid login names. Second, it allows people to determine login patterns for the valid login names. Combining this information, a hacker may attempt to break into a system while the account is normally in use, disguising their activities.

rarpd(ADMN) — This daemon replies to reverse-ARP (RARP) requests. More information is provided in Section B.2, "The RARP method, rarpd(ADMN), and the /etc/ethers file," on page B-9.

7.1.6 Starting and stopping TCP/IP

TCP/IP is started and stopped automatically when the system enters different run states. If you need to change the level of TCP/IP activity in a given run state, you can start and stop TCP/IP manually.

If necessary, briefly review run-states with students. Refer them to the **init**(ADM) manual page if they are completely unfamiliar with this concept.

7.1.6.1 Stopping TCP/IP

To stop TCP/IP from single- or multiuser-mode:

- Make sure all TCP/IP-dependent services are stopped first, including NIS and NFS
- Enter the command:

```
/etc/tcp stop
```

7.1.6.2 Starting TCP/IP

To start TCP/IP from single- or multiuser-mode, enter the command:

```
/etc/tcp start
```

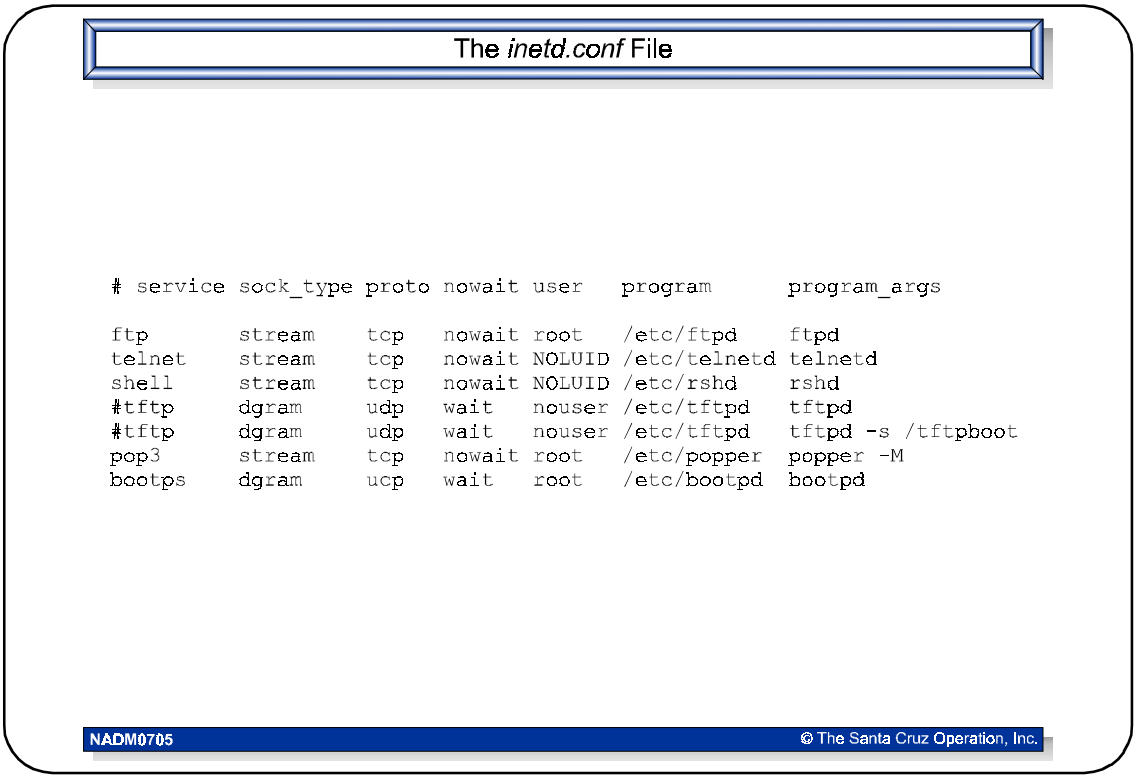
Recall that the functionality in single-user mode is limited as discussed in Section 7.1.3, “Maintenance-mode /etc/tcp activities,” on page 7-11.

The **sd**(ADM) command is required to start TCP/IP if the system is configured for the Improved or High security. To start TCP/IP, enter:

```
sd /etc/tcp start
```

Briefly review the **sd**(ADM) command for students who are unfamiliar with this command.

AV 7-5


The inetd.conf File
service sock_type proto nowait user program program_args
ftp stream tcp nowait root /etc/ftpd ftpd
telnet stream tcp nowait NOLUID /etc/telnetd telnetd
shell stream tcp nowait NOLUID /etc/rshd rshd
#tftp dgram udp wait nouser /etc/tftpd tftpd
#tftp dgram udp wait nouser /etc/tftpd tftpd -s /tftpboot
pop3 stream tcp nowait root /etc/popper popper -M
bootps dgram ucp wait root /etc/bootpd bootpd
NADM0705
© The Santa Cruz Operation, Inc.

7.2 The inetd(ADMN) Super Daemon

In the early days of the Internet, each service had its own daemon. As the number of services grew, the process table and RAM became filled with occasionally-used processes, degrading performance. The solution to this problem is **inetd**(ADMN), the Internet super daemon.

7.2.1 How inetd works

The **inetd** daemon listens for connection requests to daemons providing 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/inetd.conf* file.

AV 7-5: The *inetd.conf* File
This slide contains the first several entries from the */etc/inetd.conf* file. Walk students through each of the entries on the slide to get them familiar with the layout of the file. Be sure to mention that the value in the first field comes from */etc/services*. Have them display the file on their screen and go through the rest of the entries.

They can also look up **inetd**(ADMN) and *inetd.conf*(SFF) manual pages as you discuss the slide.

The *inetd.conf*(SFF) manual page describes the format of */etc/inetd.conf*. AV 7-5 shows a portion of a sample *inetd.conf* file that:

- Lists all the server daemons started by **inetd** (e.g., **rlogind** and **telnetd**)
- Shows several commented entries, preventing them from being started by **inetd**

For example, the **bootpd** daemon is not started by **inetd** by default because its associated entry in */etc/inetd.conf* is commented out. Have students look at this entry.

- Shows several commented entries that provide different ways for **inetd** to run daemons

For example, the **tftpd** daemon is not started by **inetd** by default. There are two possible ways to start this daemon. Because neither is used by default, they are commented. Have students look at this entry.

The */etc/inetd.conf* file can be modified using a standard text editor and the procedure outlined in Section 7.2.2 on page 7-19.

7.2.2 Configuring inetd using the *inetd.conf* file

To change the options for a server daemon that is started by **inetd**:

- Edit the */etc/inetd.conf* file.

- Force **inetd** to re-read the file. Send the SIGHUP signal (signal 1).

Exercise: Configuring inetd

Step 1. Verify that you can **telnet** to your local host, enter:

```
telnet localhost
```

Step 2. Edit the */etc/inetd.conf* file and enter a pound sign (#) in front of the line that starts **telnetd**.

Step 3. Send the **inetd** process the HUP signal, enter:

```
kill -1 ‘cat /etc/inetd.pid‘
```

Step 4. Verify that you cannot **telnet** to your local host by repeating Step 1.

Step 5. Undo the changes made in Step 2.

Step 6. Repeat Step 3.

Step 7. Repeat Step 1.

7.2.3 Enabling and disabling daemons

You might want to disable some daemons for security reasons. As illustrated in the previous exercise, you can disable daemon startup from **inetd** by editing the */etc/inetd.conf* file and commenting out the desired entry. Again, **inetd** must be sent SIGHUP to cause it to re-read the */etc/inetd.conf* file.

To enable daemon startup, add an entry in */etc/inetd.conf*, as described in the next section.

7.2.3.1 Adding new services

As your users become more sophisticated, they might require networking services started by **inetd** that do not have entries in the */etc/inetd.conf* file. You can make new entries in this file to start these services.

For example, to add a new daemon started by **inetd**, you can create an entry that is similar to this:

```
newd  stream  tcp  nowait  root  /usr/bin/newd newd  /usr/lib/newd.conf
```

The example assumes that **newd** (new daemon) is in */usr/bin/newd*, that there is (or will be) a **newd** entry in */etc/services*, and that the program takes an argument of the path name to some configuration file.

You will also add an entry for **newd** (from the first column) into the */etc/services* file. If **newd** uses port 1023, your entry in */etc/services* will be:

```
newd       1023/tcp
```

NOTE: Not all Internet service daemons are started by **inetd**. They are written to conform to the **inetd** protocols. Also, just because you can start a daemon from */etc/inetd.conf*, does not mean that you should. For example, running web servers from *inetd.conf* can have a tremendous impact on the performance of your system.

This might be a good time to do the lab in Section 7.4, "Lab: Modifying TCP/IP startup," on page 7-46.

7.2.4 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 diagnostic messages after connecting to, or receiving messages from a client, then you can use **telnet** to send the server messages.

In addition to a resolvable host name or dotted-decimal notation IP address, **telnet** can also connect to a port number. For example, to connect your **telnet** client to the **finger** daemon, enter:

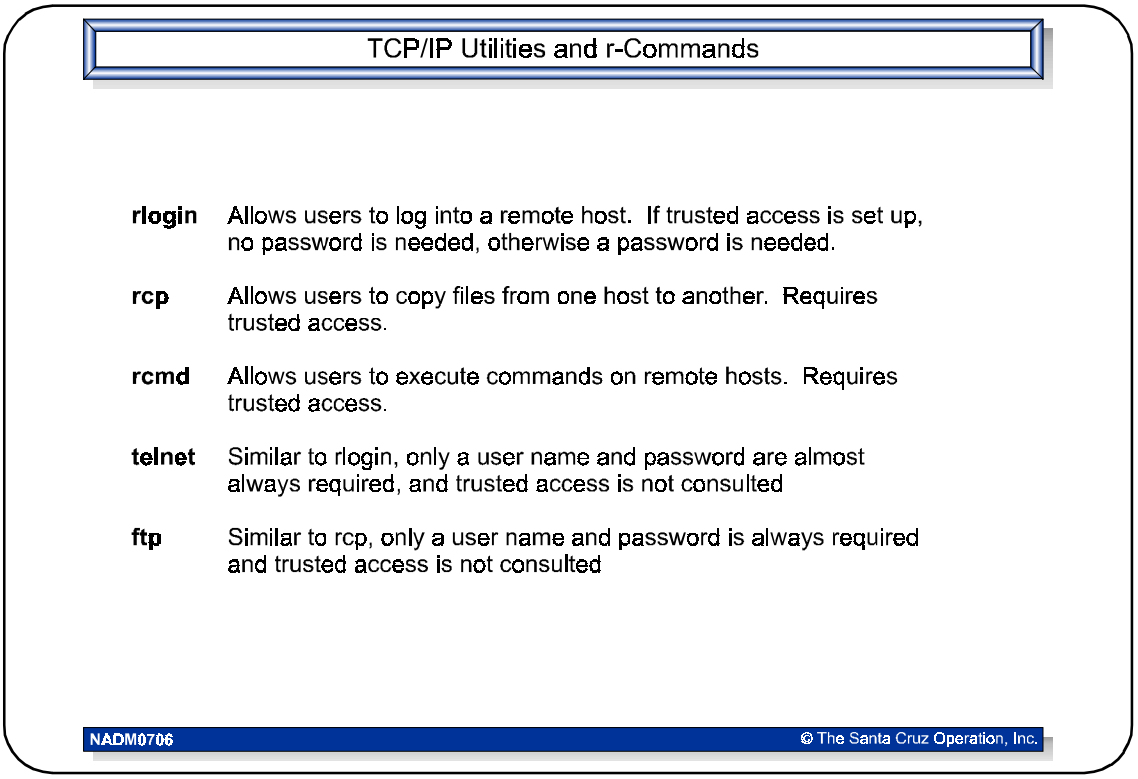
```
telnet localhost 79
Connected to localhost.
Escape character is '^]'.
```

After connecting to the finger daemon you can enter **finger** command arguments. For example, to display information about **root**, enter:

```
root
Login: root              Name: Superuser
Directory: /             Shell: /bin/sh
On since Mon Jan 10 19:06 on tty02, idle 3:13
Mail last read Tue Jan 11 00:28:03 1998
No Plan.
Connection closed by foreign host.
```

You might want to experiment with connecting to other daemons, such as SMTP on port 25 to demonstrate how easy it is to forge an e-mail document. You might further discuss ways to detect a forgery; look at the time-stamp on an e-mail message and compare that with login or process accounting records of the supposed sender. If the supposed sender was not logged in or running an e-mail program near the time of the time-stamp, the e-mail is most likely a forgery.

AV 7-6



Instructor Guide

7.3 Configuring Trusted Access

Trusted access allows users to access the enterprise network in a secure and convenient manner.

For example, if trusted access is not configured for the **rlogin**(TC) command:

1. It prompts 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.

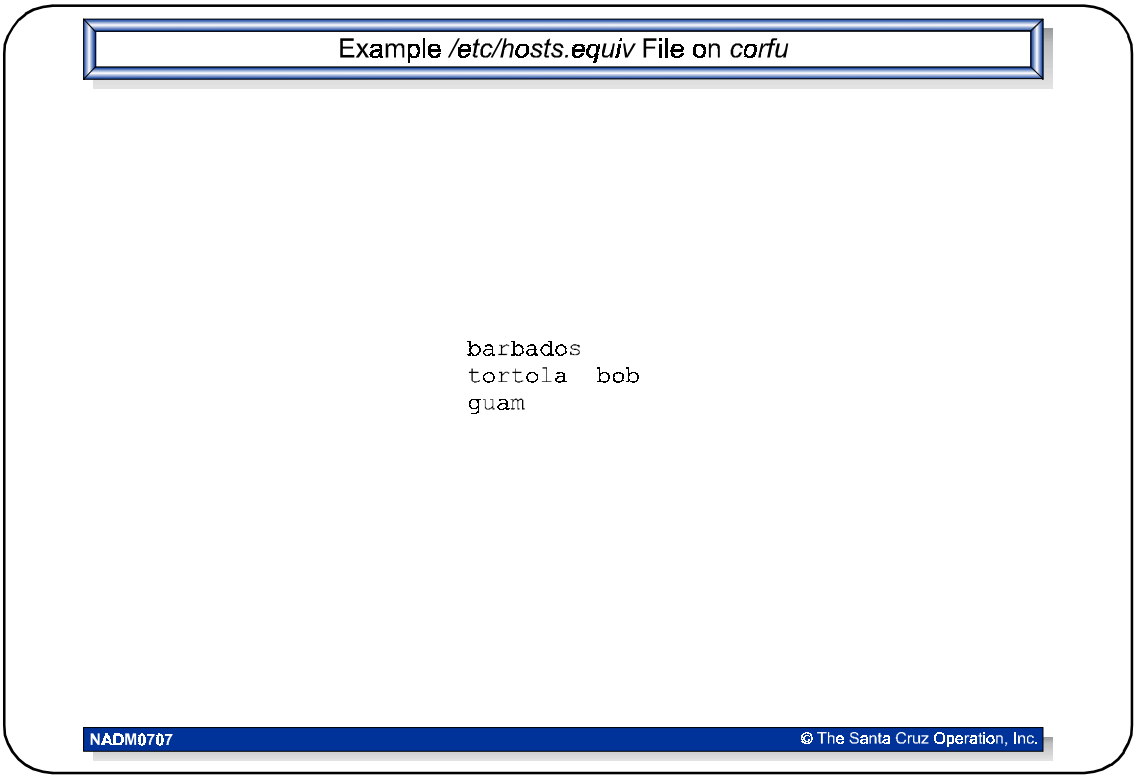
If trusted access is set up, no password is necessary for **rlogin**. If trusted access is not set up, the **rcmd**(TC) and **rcp**(TC) commands do not work at all.

WARNING: Regardless of whether you have configured trusted access, if you have a user account with no password, a user on another system with the same login name can use all **r**-commands to access your system without a password.

As a network administrator, your responsibilities include administration of trusted access and *user equivalence*, another network attribute that is explained in the next sections. You can configure trusted access at the host or user level, or both.

AV 7-6: TCP Utilities and r-Commands
Use this slide to remind students what the illustrated commands do. Poll your students. If they are not familiar with the r-commands, use the job aid to teach them how they work.

AV 7-7


Example /etc/hosts.equiv File on corfu
barbados
tortola bob
guam
NADM0707
© The Santa Cruz Operation, Inc.

7.3.1 Configuring trusted host access

Trusted host access indicates that one host trusts the users on another host. When configured, these commands can be used without specifying a password:

- **rlogin**
- **rcp**
- **rcmd**

Explain the exception of using the **-l** flags to **rlogin** and **rcmd** and the `user@` part of **rcp** pathnames.

You must be **root** to configure trusted access at the host level. This configuration consists of creating or editing the */etc/hosts.equiv* file. Each entry in */etc/hosts.equiv* specifies the trusted remote system. AV 7-7 illustrates the format of entries in an example */etc/hosts.equiv* file.

AV 7-7: Example */etc/hosts.equiv* File on *corfu*
Use this slide to demonstrate and define the possible form of entries in this file.

If the entries in AV 7-7 exist on a machine called *corfu*, they have this meaning:

`barbados`	All non-**root** users on *barbados* with the same user name as those on this machine (*corfu*) are trusted. In other words, **jane** on *barbados* can **rlogin**, **rcp**, and **rcmd** to *corfu* as **jane** without specifying Jane's password on *corfu*.
`tortola bob`	Allows **bob** on *tortola* to log in as user **bob** on this machine (*corfu*). WARNING: This entry allows **bob** on *tortola* to log in as any user on *corfu*, except **root**. This can be dangerous!

The **root** account can never be equivalenced by an entry in the */etc/hosts.equiv* file. Refer to *hosts.equiv*(SFF) 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.

Explain why you should use fully-qualified host names.

7.3.2 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**
- **rcmd**
- **rcp**

Unlike trusted host access, trusted user access can be created for users on different machines with different login names.

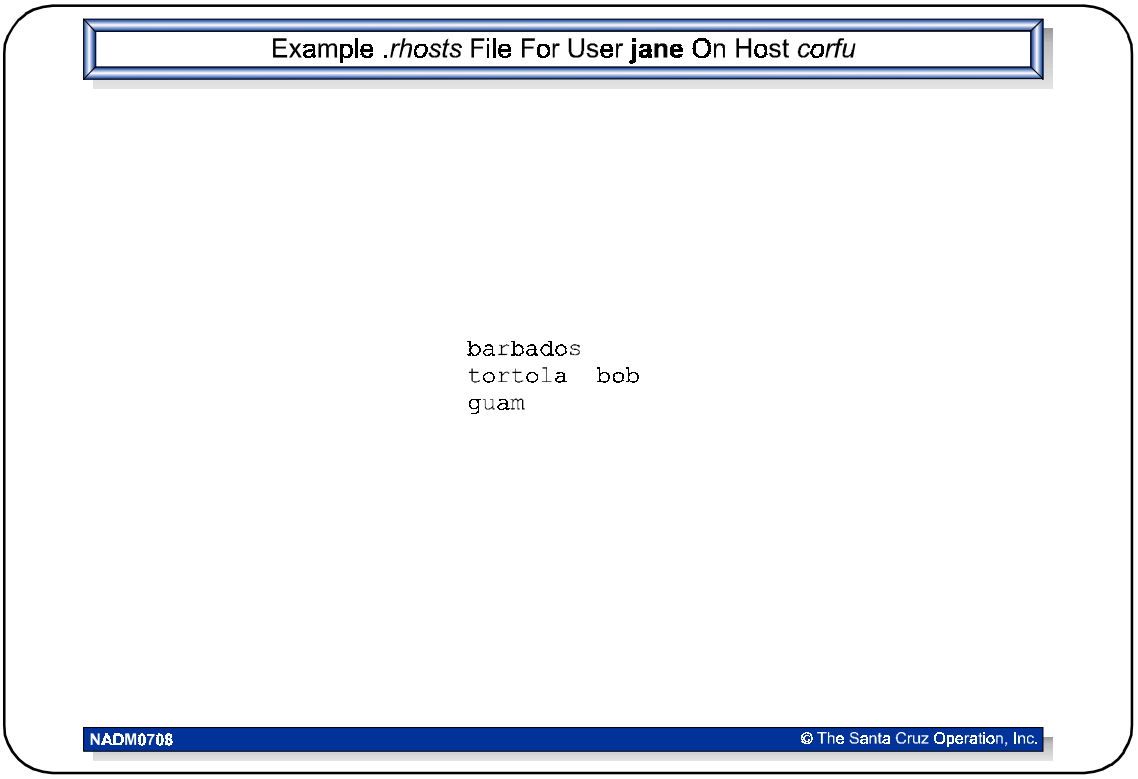
You can configure trusted user access for your users, or they can do it for themselves. The configuration consists of:

1. Creating or modifying a file called *.rhosts* (note the leading dot) in a user's home directory.
2. Setting the ownership so that the file is owned by either **root** or the owner of the home directory.
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.

Point out that this may be problematic for the unusual machine that has multiple users using the same home directory.

AV 7-8


Example .rhosts File For User jane On Host corfu
barbados
tortola bob
guam
NADM0708
© The Santa Cruz Operation, Inc.

AV 7-8: Example *.rhosts* File for User jane on Host corfu
Use this slide to illustrate the use of the *.rhosts* file.

The format of entries in the *.rhosts* file is the same as those in the */etc/hosts.equiv* file. However, the semantics are somewhat different. If the entries shown in AV 7-8 are put in the home directory of the user **jane** on the machine *corfu*, they have the following meaning:

`barbados`	User **jane** on *barbados* is trusted by the user **jane** on this machine (*corfu*). In other words, **jane** on *barbados* can **rlogin**, **rcp**, and **rcmd** to *corfu* as **jane** without specifying Jane's password on machine *corfu*.
`tortola bob`	User **bob** on *tortola* is trusted by the user **jane** on this machine (*corfu*). In other words, **bob** on *tortola* can **rlogin**, **rcp**, and **rcmd** to this machine (*corfu*) as **jane**, using the appropriate arguments to those commands, without specifying Jane's password.

Ask students, "if the *.rhosts* file must exist in a user's home directory, and **root** cannot be made equivalent via */etc/hosts.equiv*, how can **root** be made equivalent?" Answer: by creating a file called *.rhosts* in the root (*/*) directory.

Exercise: Setting up trusted access

For this exercise, you will need a partner.

Step 1. Log in as **guest** on your system. Your instructor will give you the password for the **guest** account.

Step 2. Remotely log into your partner's system using this command:

`rlogin` *`partner's_system_name`*

Are you prompted for a password?

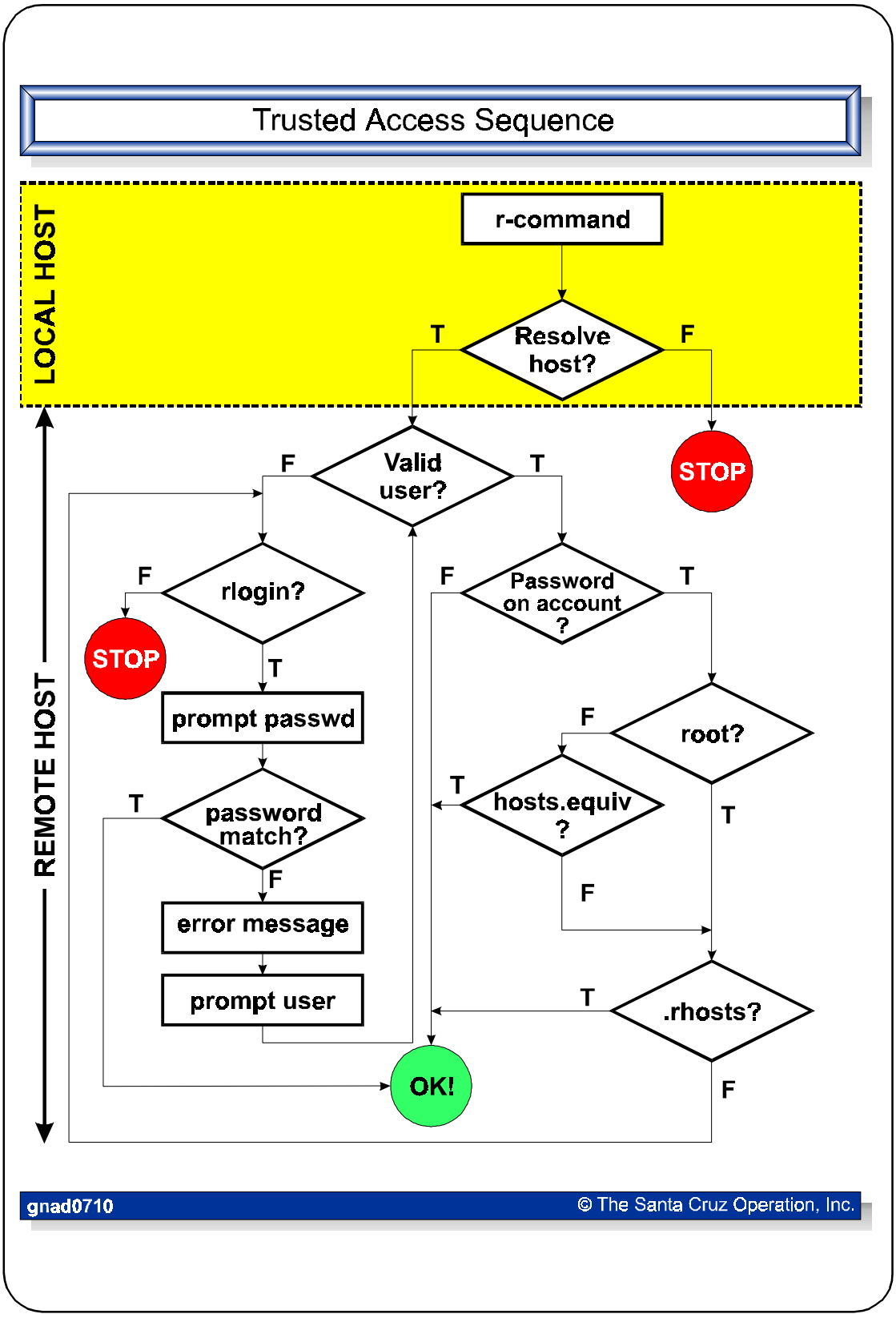
Step 3. After you have logged in, edit or create a *.rhosts* file in the home directory of **guest** on your partner's machine and add a line with just the name of your system.

Step 4. 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 7-9


Trusted Access Sequence
LOCAL HOST
r-command
Resolve host?
T
F
STOP
Valid user?
F
T
REMOTE HOST
rlogin?
F
STOP
T
prompt passwd
password match?
T
F
error message
prompt user
Password on account ?
F
T
root?
F
T
hosts.equiv ?
T
F
.rhosts?
T
F
OK!
gnad0710
© The Santa Cruz Operation, Inc.

7.3.3 The trusted access sequence

AV 7-9: Trusted Access Sequence
Use this slide to walk students through this procedure, which determines whether a user is trusted.

Whenever an **rlogin**, **rcmd**, or **rcp** command is executed, the systems involved goes through a set procedure to determine whether trusted access is configured. AV 7-9 shows this procedure.

7.3.3.1 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, this error message is displayed:

   host_name: Host name lookup failure

4. If an address for the remote host can be resolved, the **r**-command connects through the TCP/IP transport to the remote host.

7.3.3.2 Trusted access on the remote host

Note the use of the term password database instead of the */etc/passwd* file. This is necessary. After all, the password file might be under NIS control. You might want to indicate that user names and passwords can exist outside the */etc/passwd* file.

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 **rcmd** and **rcp** commands fail, and the **rlogin** command prompts for a password (that is how the login program functions), and then fails.
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 user accounts without 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.

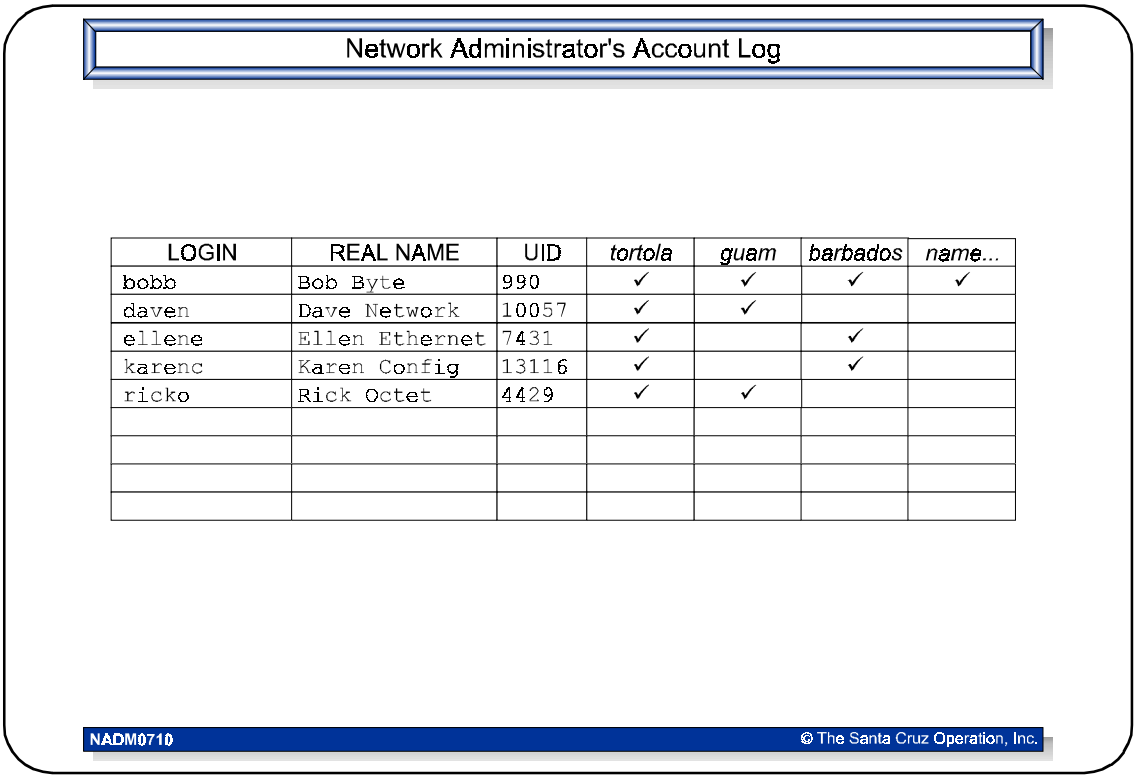
   If */etc/hosts.equiv* contains the name of the local host, the **r**-command executes.

6. If the account name is **root**, or if the */etc/hosts.equiv* check fails, 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, the **r**-command executes.

7. If the *.rhosts* file contains the desired entry, then the **r**-command executes.

Failure so far means that the **rcmd** and **rcp** commands will fail. If the **r**-command invoked was **rlogin**, it prompts for the remote user's password, and only succeeds if the correct password is given.

AV 7-10


Network Administrator's Account Log
LOGIN
REAL NAME
UID
tortola
guam
barbados
name...
bobb
Bob Byte
990
daven
Dave Network
10057
ellene
Ellen Ethernet
7431
karenc
Karen Config
13116
ricko
Rick Octet
4429
NADM0710
© The Santa Cruz Operation, Inc.

7.3.4 Overview of user equivalence

Section 7.3.3, “The trusted access sequence,” on page 7-35 showed that trusted access is based upon user names. What would happen if:

- Two people on different machines had the same user name
- Trusted host access was established between the two hosts

The answer is that 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 AV 7-10, such a log should contain:

- User account name and real name
- User ID
- Systems on which a given user has an account

AV 7-10: Network Administrator’s Account Log
Use this slide to show students how they can track on which systems a given user has an account.

A simple example of user equivalence is illustrated with two users on two systems:

- You have two people, John and Jane.
- Each person has accounts on both machines, *corfu* and *barbados*, with the same account names and user IDS:

corfu	**john**, 201 **Jane**, 202
barbados	**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 among machines between:

- Group names
- Group IDs
- Group members

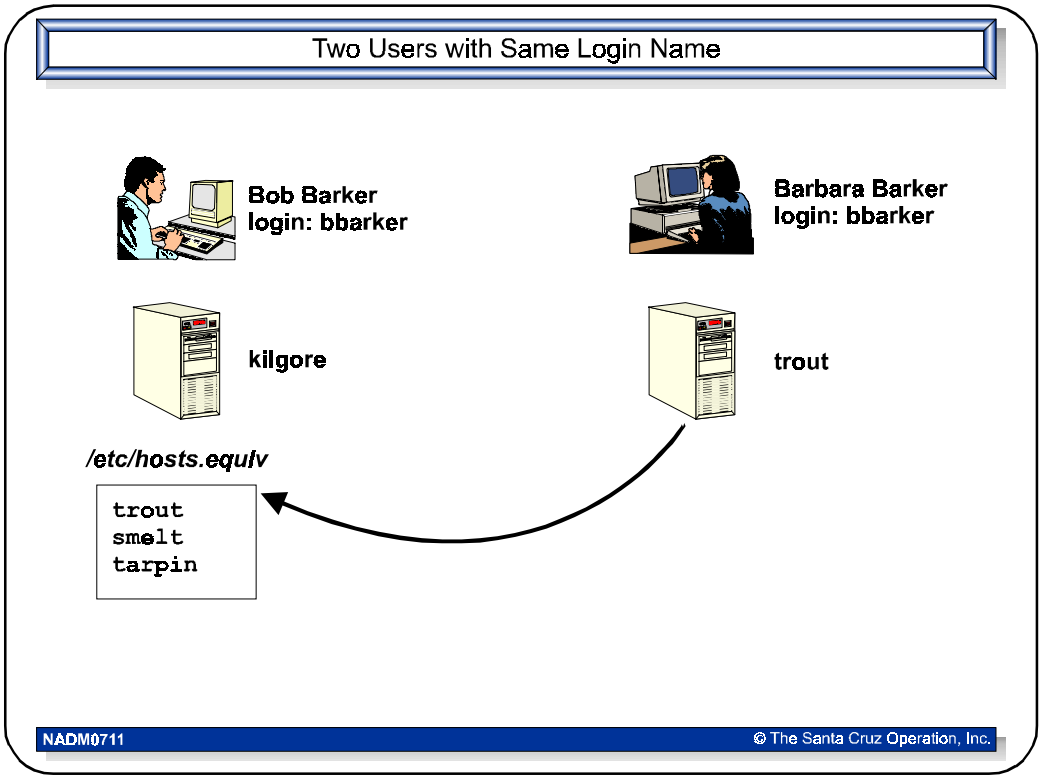
You need to establish user equivalence if you expect to use trusted access or NFS on your hosts. It is strongly recommended that you establish unique account names, UIDs, and GIDs, which are replicated to each system on the network.

The maintenance of user equivalence is made much easier with the Network Information Service (NIS). For more about using NIS, see **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Chapter 9: Configuring the Network Information Service (NIS)**.

Instructor Guide

The motivation for students to establish user equivalence on all systems in their network is clarified in the next sections.

AV 7-11


Two Users with Same Login Name
Bob Barker
login: bbarker
Barbara Barker
login: bbarker
kilgore
trout
/etc/hosts.equiv
trout
smelt
tarpin
NADM0711
© The Santa Cruz Operation, Inc.

7.3.4.1 Network access problems

This 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 a password.
- 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 whether Bob and Barbara have the same UID.

Reinforce with students that trust is based on user names, not user IDs.

AV 7-11: Two Users with Same Login Name
Use this slide to illustrate potential problems with trusted access without first setting up user equivalence.

You might have to do a bit of a forward reference to what NFS is, but do not go much beyond saying that it is like mounting a filesystem from another machine on your local machine's file tree, as though it were a local disk. Indicate that you will talk more about it in Appendix A: "NFS."

This 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 shows him as the owner of the files.

For this example, it does not matter whether Bob and Barbara have the same account name.

For NFS and many other file-sharing software packages, the UID is the thing that must be the same, regardless of the login names, unlike the **rcmd**, **rcp**, and **rlogin** commands.

7.3.5 Changing an account's UID and login name

If you find that you have a conflict like one of those described, you need to clearly analyze the problem and carefully create a plan to resolve the conflicts. You can use the **usermod**(ADM) command to modify an account's UID and login name.

For example, to change the login name of user **bbarker** to **bobb**, enter:

```
usermod -l bobb 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 accounts database. You might need to change file and directory ownership.

1 Lesson Summary

This lesson discussed the many issues related to TCP/IP administration including startup, shutdown, configuration, the TCP/IP daemons and their function.

It also described how trusted access and user equivalence can make your network more secure and easier to use.

Review questions

1. Which file lists the server daemons started by **inetd**(ADMN)?

2. Which signal do you send to **inetd**(ADMN) to tell it to re-read its configuration file?

3. Name three ways that server daemons can be started.

4. SWhich files are used to configure trusted access for hosts and users?

5. Will trusted access to an account on a remote machine without a password always fail? Explain.

6. What must be configured for either **rcp** or **rcmd** to function?

7.4 Lab: Modifying TCP/IP startup

In this lab, you will modify the behavior of TCP/IP startup by adding the **scohttpd** daemon to */etc/inetd.conf*.

The **scohttpd** daemon is a document server used by **scohelp**(XC) and **man**(C) to display online documentation. By default, this daemon is started from */etc/rc0.d/P93scohttp* when the system enters single-user mode, or */etc/rc2.d/P93scohttp* when entering multiuser mode.

Note that when **scohttpd** is run from **inetd**, it must fork a process and parse its configuration files for each help request it receives. This is an increase in CPU load as compared with the default standalone **scohttpd** configuration. When running standalone, the configuration files are parsed once at startup.

While this example is useful for demonstrating additions to */etc/inetd.conf*, it might not be a desirable configuration for your system.

Procedure

To configure **inetd** to start **scohttpd**:

Step 1. Verify that **scohttpd** is running. Enter:

```
man man
```

The **man**(C) manual page is displayed.

Step 2. Edit the */etc/services* file to add the following line (if it does not already exist):

```
scohelp 457/tcp
```

Step 3. Edit the */etc/inetd.conf* file to add the following line:

```
scohelp stream tcp nowait root /var/scohttp/scohttpd scohttpd
-d /var/scohttp
```

This is entered on a single line in */etc/inetd.conf*.

Step 4. Edit the file, */var/scohttp/conf/scohttpd.conf*. Change the directive `ServerType` from `standalone` to `inetd`. For example:

```
ServerType inetd
```

Step 5. Stop the **scohttp** daemon. Enter:

```
scohttp stop
```

Step 6. Verify that **scohttpd** is not running using the **man** command:

```
man man
man : can't connect to scohttp
man: man not found
```

Step 7. Edit */etc/profile* and add these lines:

```
MANPORT=457
MANSERVER=`hostname`
export MANPORT MANSERVER
```

Step 8. Execute */etc/profile*

```
. /etc/profile:
```

Step 9. Signal the **inetd** process:

```
kill -1 'cat /etc/inetd.pid'
```

Step 10. Verify that scohttp is now running by running the **man** command:

```
man man
```

2 Relevant Commands and Files for TCP/IP Administration

The following commands and files were presented in this lesson.

Relevant commands

rlogin(TC)	UNIX-specific remote login
gated(ADMN)	A multi-protocol routing daemon
ifconfig(ADMN)	Network interface configuration
inetd(ADMN)	Internet super daemon
irdd(ADMN)	The Internet Route Discovery Daemon
lpd(ADMN)	Remote line-printer daemon
rarpd(ADMN)	Reverse ARP daemon
rcmd(TC)	UNIX specific remote command
rcp(TC)	UNIX specific remote copy
routed(ADMN)	A RIP-only routing daemon
slink(ADMN)	Configures STREAMS and socket
snmpd(ADMN)	Simple network management protocol daemon
syslogd(ADM)	Logs system messages to */usr/adm/syslog* by default
telnet(TC)	TCP/IP standard remote login
uname(C)	Get or set a machine's name and other properties
xntpd(ADMN)	Standard time daemon

Relevant files

~/.rhosts	Configuration file for user trusted access
/usr/adm/syslog	Default location for output from **syslogd**(ADM)
/etc/gated.conf	Configuration file for **gated**(ADMN)
/etc/hosts.equiv	Configuration file for host trusted access
/etc/inetd.conf	Configuration file for **inetd**(ADMN)
/etc/xntpd.conf	Configuration file for **xntpd**(ADMN)
/etc/strcf	STREAMS configuration file used by **slink**(ADMN)
/etc/tcp	Script to start and stop TCP/IP

7.5 Answers to Review Questions

This section provides suggested answers to review questions.

1. The */etc/inetd.conf* file lists the server daemons started by **inetd**(ADMN).

2. Send **inetd**(ADMN) signal 1, the HUP signal, to tell it to re-read its configuration file.

3. Server daemons can be started in any of the following ways:

   - Through an entry in a system startup script (such as */etc/rc0.d/P93scothhp*)
   - Through an entry in */etc/tcp*
   - Through an entry in */etc/inetd.conf*

4. The */etc/hosts.equiv* and *.rhosts* files in the user's home directory are used to configure host and user trusted access, respectively.

5. 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.

6. Trusted access must be configured for either **rcp** or **rcmd** to function. The shell entry in */etc/inetd.conf* must also be configured.

8

Domain Name System (DNS) Configuration

Instructor Guide

Lesson 8

Domain Name System (DNS) Configuration

Suggested lesson duration

This lesson should take approximately 4 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

Ensure the classroom machines are all functioning properly on the network using */etc/hosts* for resolving IP addresses and host names. Remember to isolate the classroom machines from the AEC business network, if necessary.

Instructor preparation

You should be thoroughly familiar with domain names, the domain name tree, resolvers, and the SCO implementation of BIND. Specifically, you should understand the SCO specific */etc/resolv.conf* file entries.

You must also be thoroughly familiar with name server configuration file entries including both forward and reverse lookup entries.

NOTE: SCO OpenServer Release 5.0.5 has updated to BIND version 8.1.1. This uses a new configuration file, *named.conf*. However, the old-style *named.boot* file can continue to be used. If found, **named** transparently creates a new *named.conf* file using the information in *named.boot*. There are also standalone utilities to create *named.conf* from *named.boot*. See the README file in */etc/named.d* for more details.

This lesson continues to teach configuration using the old-style *named.boot* for these reasons:

- The new version of named is completely compatible.
- The *named.boot* file is still used in other SCO operating systems, including older versions of OpenServer Release 5.
- The DNS configuration chapter in the Networking Guide has not been updated to reflect BIND 8.1.1; the *named.boot* manual page still exists, with a reference to *named.conf*(SFF).
- Use of *named.boot* is common throughout the industry at this time.

If you direct students to use the sample files in */etc/named.d* when creating the other DNS configuration files (such as *named.local*, *named.rev*, etc.), they will probably notice that there are no *named.boot* sample files. The old-style sample *named.boot* files are also still referred to in the documentation.

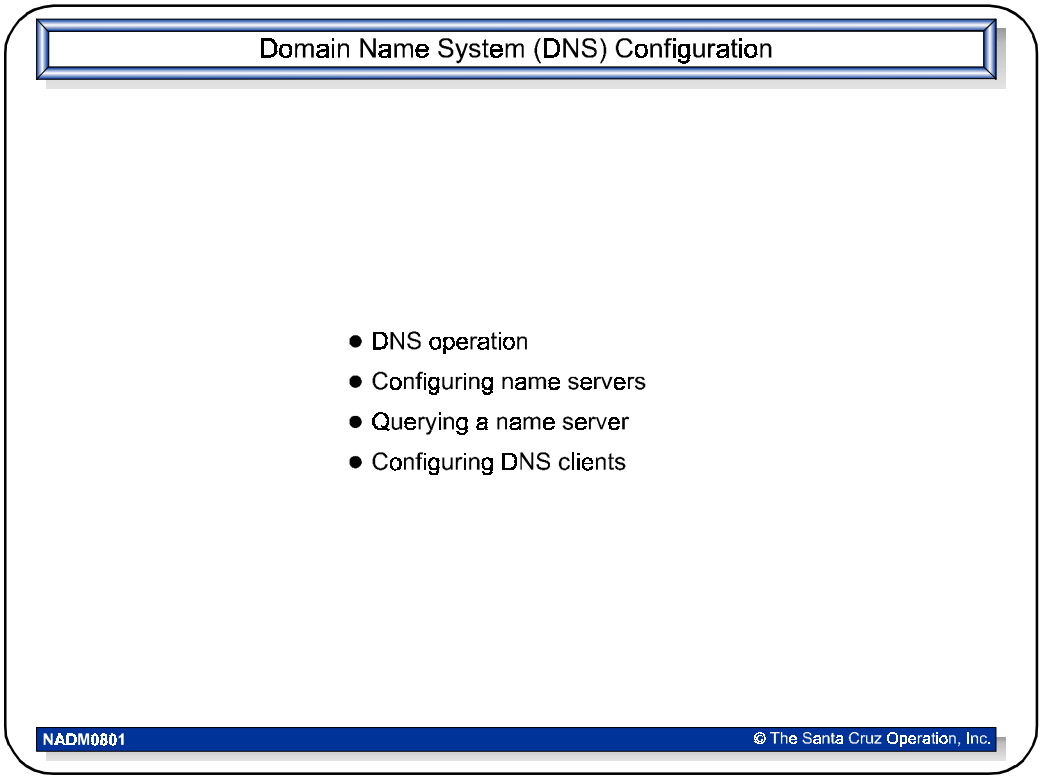
Mention of *named.conf* is made in Section 8.2.1, "The named.boot file," on page 8-23. If necessary, expand on the rationale for using *named.boot* as the basis for this lesson.

A discussion of DNS can be found in **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Domain Name Service.** A good reference book is:

Albitz, Paul & Cricket Liu, ***DNS and BIND***, 2nd Edition, Sebastopol, CA: O'Reilly & Associates, 1997, ISBN: 1-56592-236-0

NOTE: Throughout this lesson, the same example network is used for consistency. The example network is in the *somedom.com* domain with a network address of 200.100.50 and host name and host address pairs of: *thatch*/1, *guam*/2, *atlantis*/3, *gateway*/4, *roscoe*/5, and *cush*/6. The network administrator has also agreed to serve as a secondary name server for the *otherdom.com* domain, whose network address is 200.150.75. Keep this in mind as you work through the examples with students. You might also want to state this assumption to the students at the beginning of the lesson.

AV 8-1


Domain Name System (DNS) Configuration
DNS operation
Configuring name servers
Querying a name server
Configuring DNS clients
NADM0801
© The Santa Cruz Operation, Inc.

Introduction

Lesson 2: “TCP/IP Concepts” discussed the resolution of IP addresses to host names using the */etc/hosts* file. Another alternative for host name resolution is the *Domain Name System* (DNS).

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.

NOTE: There are many chicken-and-egg issues when presenting this topic. Because of this, there are many forward references to record types and fields in those records. Tell students at the beginning and throughout the lesson that the actual record formats are discussed later. Give students incidental information (such as what an A record is) when they are first introduced, but do not get specific.

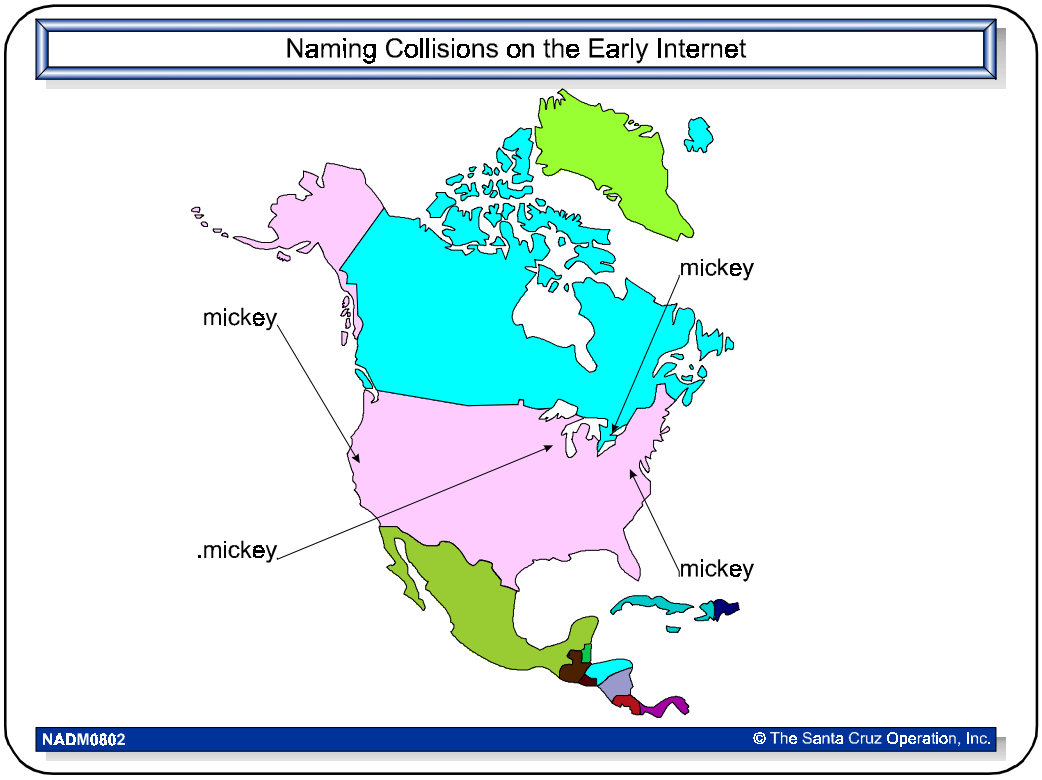
Objectives

This lesson describes how to set up DNS on machines running SCO OpenServer Release 5. By the end of this lesson, you will be able to:

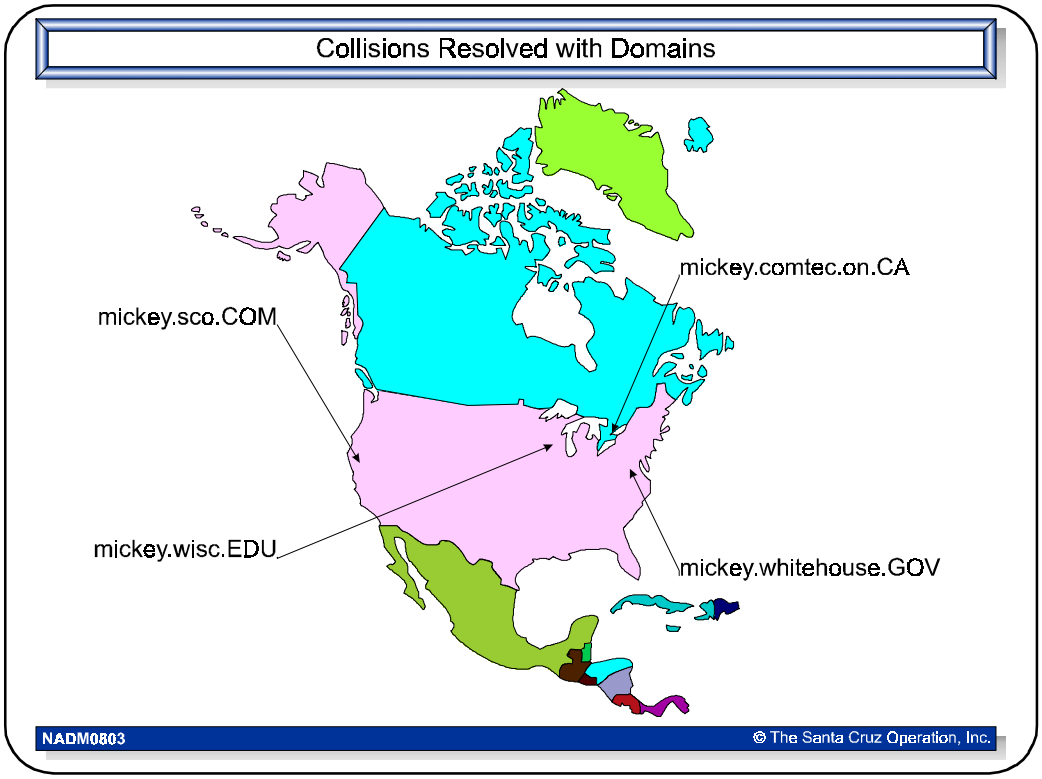
- Describe what DNS is and how it works
- Create a database of DNS information for a domain using the various DNS record types in standard resource record (SRR) format
- Configure a primary, secondary, and caching-only name server
- Configure a DNS client
- Query a name server’s database

AV 8-1: Domain Name Service (DNS) Configuration
This first slide describes the objectives covered in this lesson.

AV 8-2



AV 8-3



Instructor Guide

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 became the */etc/hosts* file.

When the Internet grew to hundreds of machines, several problems with this approach emerged:

- There were occasional collisions when two sites on the Internet chose the same machine name and someone had to decide who got which address.
- As the number of systems being added to the Internet grew, maintaining the */etc/hosts* file became an unmanageable task.
- The file became so large that a linear search through the */etc/hosts* file took a significant amount of time.

AV 8-2: Naming Collisions on the Early Internet
This slide illustrates a simplified version of the early Internet. It has just a few sites; Santa Cruz, CA; Madison, WI; Washington, DC; and Toronto, Ontario. In this scenario, people in these sites all want to add a host called *mickey* to the Internet. Use this slide to illustrate how this could happen and, given a flat name space, why it was a problem.

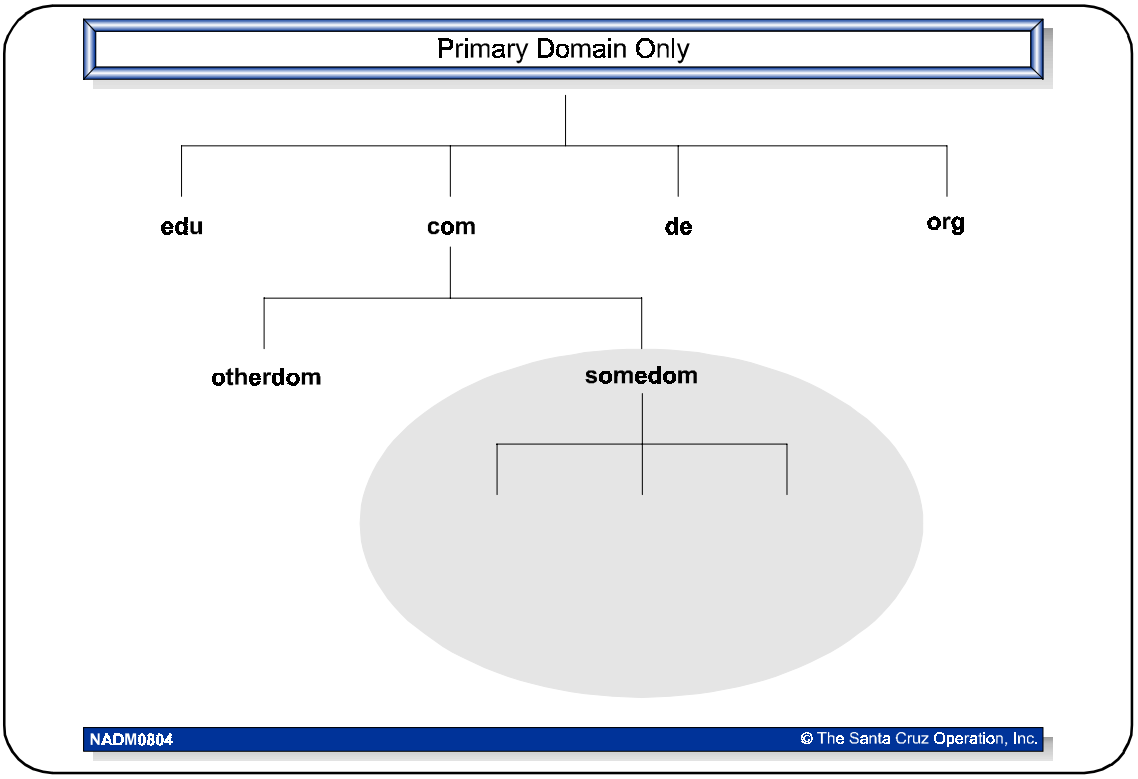
AV 8-3: Collisions Resolved with Domains
Use this slide to point out that with DNS, each site can still have a host named *mickey* because each name exists within its own domain. Do not get too bogged down in the syntax. This is a good time to discuss the definition of the term *domain*. Start with a dictionary definition, then apply the definition to machine names.

The solution eventually implemented was the Domain Name System (DNS):

- DNS was proposed by the University of California at Berkeley and called the Berkeley Internet Name Domain (BIND) service.
- The daemon that implements BIND on SCO systems running the name server is called **named**(ADMN), the name daemon.

  The **named** daemon listens for requests from clients anywhere on the Internet to resolve names and IP addresses. Additional functionality is covered later in this lesson.

AV 8-4


Primary Domain Only
edu
com
de
org
otherdom
somedom
NADM0804
© The Santa Cruz Operation, Inc.

8.1.1 DNS server types

DNS supports these server configuration types:

- Primary name servers (PNS)
- Secondary name servers (SNS)
- Caching-only name servers

8.1.1.1 Primary name servers (PNS)

In this course, the terms primary name server and secondary name server are abbreviated as PNS and SNS to be more concise. These acronyms are not commonly used.

A primary name server is *the* authority for one or more portions of the domain tree. The rest of the Internet contacts 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 the example 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 8-4: Primary Domain Only
This slide shows the name tree with the *somedom.com* domain shaded indicating that a single name server is responsible for that entire part of the tree.

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 is to set up a secondary name server (SNS).

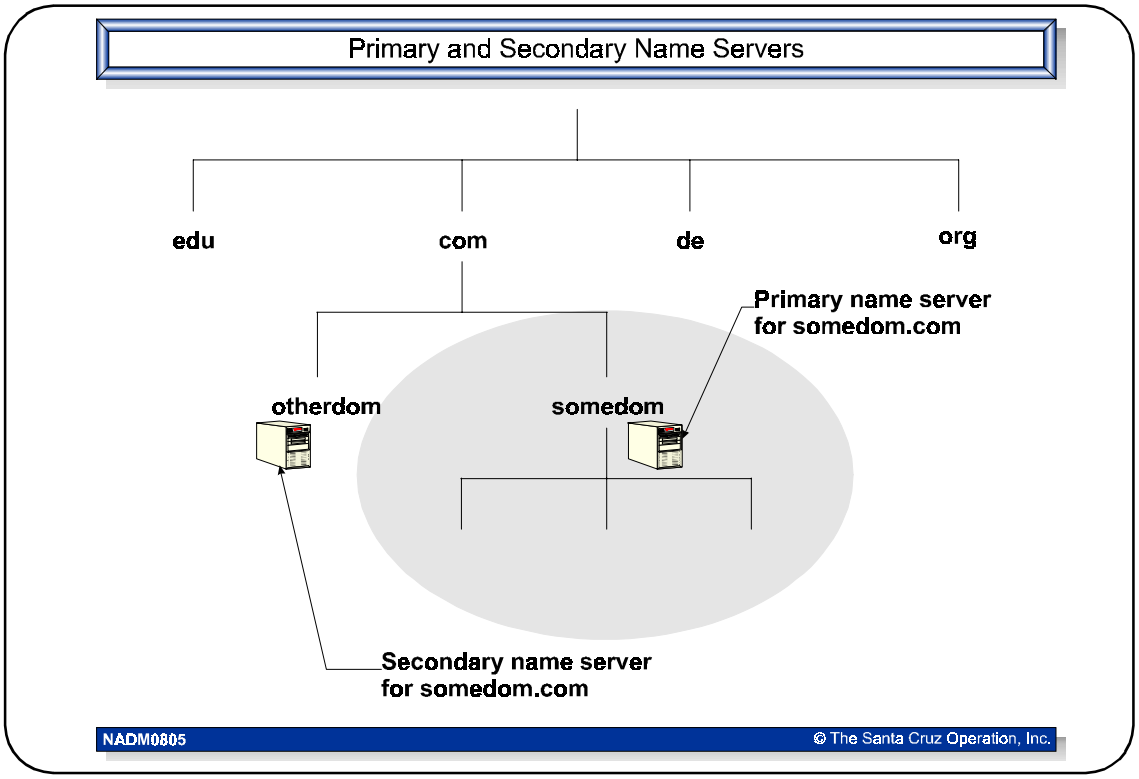
Remind students that the area of the domain tree that a name server has authority over is called its *zone of authority* or *zone*.

Although a zone can have only one primary name server, there can be multiple secondary name servers for a zone. These secondary name servers can be queried by Internet clients 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 being supplied. 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.

AV 8-5


Primary and Secondary Name Servers
edu
com
de
org
Primary name server
for somedom.com
otherdom
somedom
Secondary name server
for somedom.com
NADM0805
© The Santa Cruz Operation, Inc.

AV 8-5: Primary and Secondary Name Servers
This slide shows two name servers, a primary and a secondary, both serving information about the *somedom.com* domain.

This sequence illustrates 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 an 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 includes a serial number and indicates when it was last validated.

3. Attempts to contact the PNS to see if the serial (version) number is still the same. If repeated attempts to contact the PNS fail, the data is marked as invalid (**named** on the SNS will not serve it to clients) and **named** on the SNS might give up trying to contact the PNS.

4. Asks the PNS for a copy 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:

retry How often to repeat attempts to contact the PNS

minimum How long to keep the data

expire When to stop trying to contact the PNS

These parameters are stored in the primary name server's database and kept by the SNS whenever a successful contact is made.

Students want to know where these parameters are stored. Refer to Section 8.3.5, "The SOA record and `$include`," on page 8-55.

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 reduces network traffic and provides faster 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 participate in 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 rather than 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. Alternately, if a caching-only name server has previously been configured, the configuration files must be removed before it can be configured as another type of name server.

8.1.2 How queries are answered

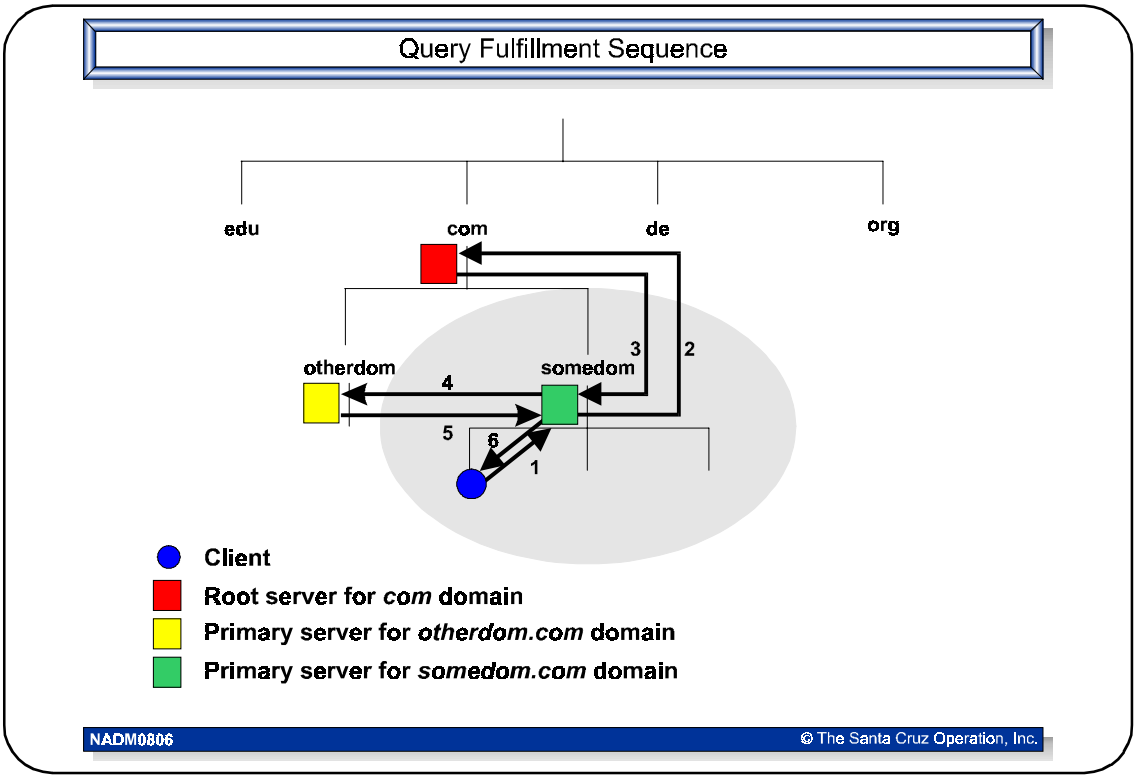
Before going into the details of setting up name servers and clients, it is useful to understand how queries are processed. There are two types of name resolution:

By *iteration* Name servers usually perform name resolution by iteration. A query is iterative when the name server conducts its own search of the domain tree.

By *recursion* Clients perform name resolution by recursion. A query is recursive when the client passes on responsibility for resolving a name to a server and waits for the answer.

The sequence shown in AV 8-6 on page 8-18, and described below, illustrates how a query for information about a machine outside the zone of the client is typically handled. In this example, the client is doing a recursive query and the name server is doing resolution by iteration.

AV 8-6



Instructor Guide

AV 8-6: Query Fulfillment Sequence
This slide shows the flow of information indicated in the sequence below, from a client to a root-level server to the primary server in the target's zone, back to the primary in our zone, and finally back to the client. The numbers next to the lines in the slide correspond to the steps below. The arrow heads indicate which machine is doing the asking or replying. Walk students through this sequence.

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*.

The TCP/IP routine to do this is **gethostbyname**(SLIB). It might also be useful to mention this in passing.

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*.

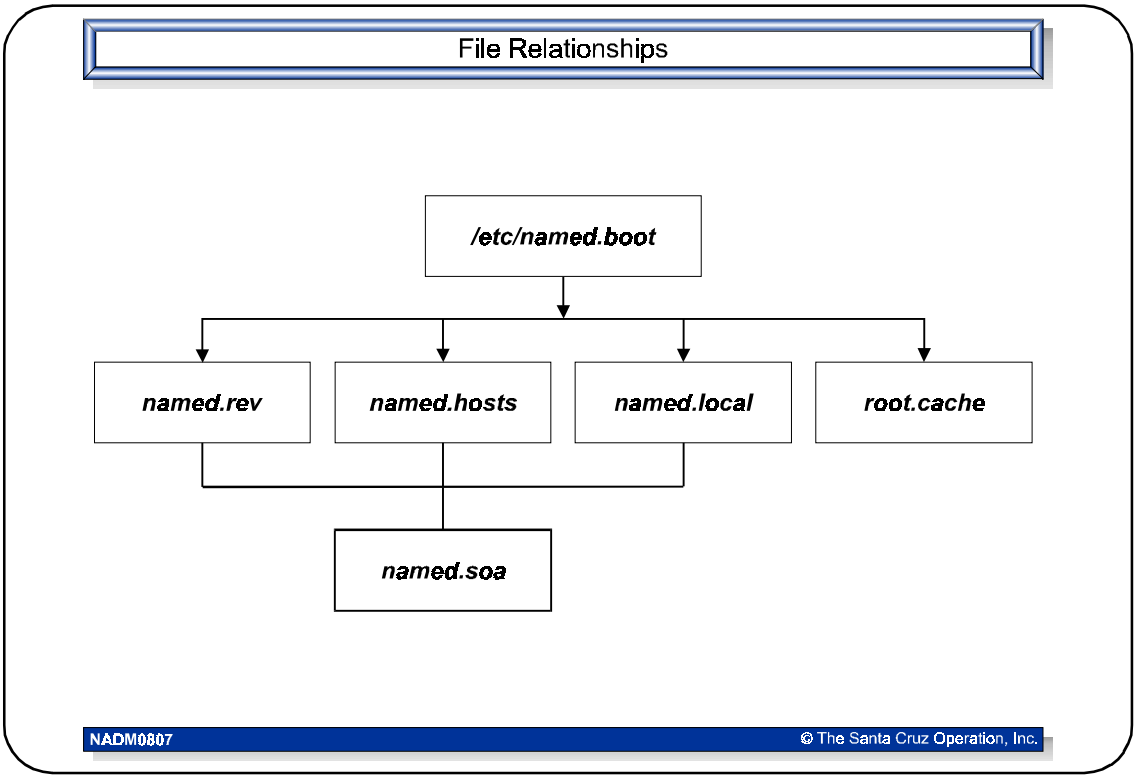
5. *dns.otherdom.com* sends a reply with the IP address of *remote.otherdom.com*.

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



Instructor Guide

8.2 DNS Files

So far, this lesson has described DNS at a conceptual level. To configure DNS, you must understand each of the components that make up the system.

Configuring DNS can be quite complex on large, sophisticated networks that require all types of servers—primary, secondary, forwarders, stubs, and so on. However, in its simplest form, DNS configuration involves these tasks:

- Creating a primary name server
- Configuring hosts as DNS clients

This set of configuration files must be created to configure a primary name server:

/etc/named.boot	Configures the **named**(ADMN) daemon on start up
/etc/named.d/named.hosts	Translates host names to IP addresses.
/etc/named.d/named.rev	Translates IP addresses to host names
/etc/named.d/named.local	Translates IP address 127.0.0.1 to *localhost*
/etc/named.d/root.cache	Contains information about the name servers for the root domain
/etc/named.d/named.soa	An include file containing a record that defines the zone of authority

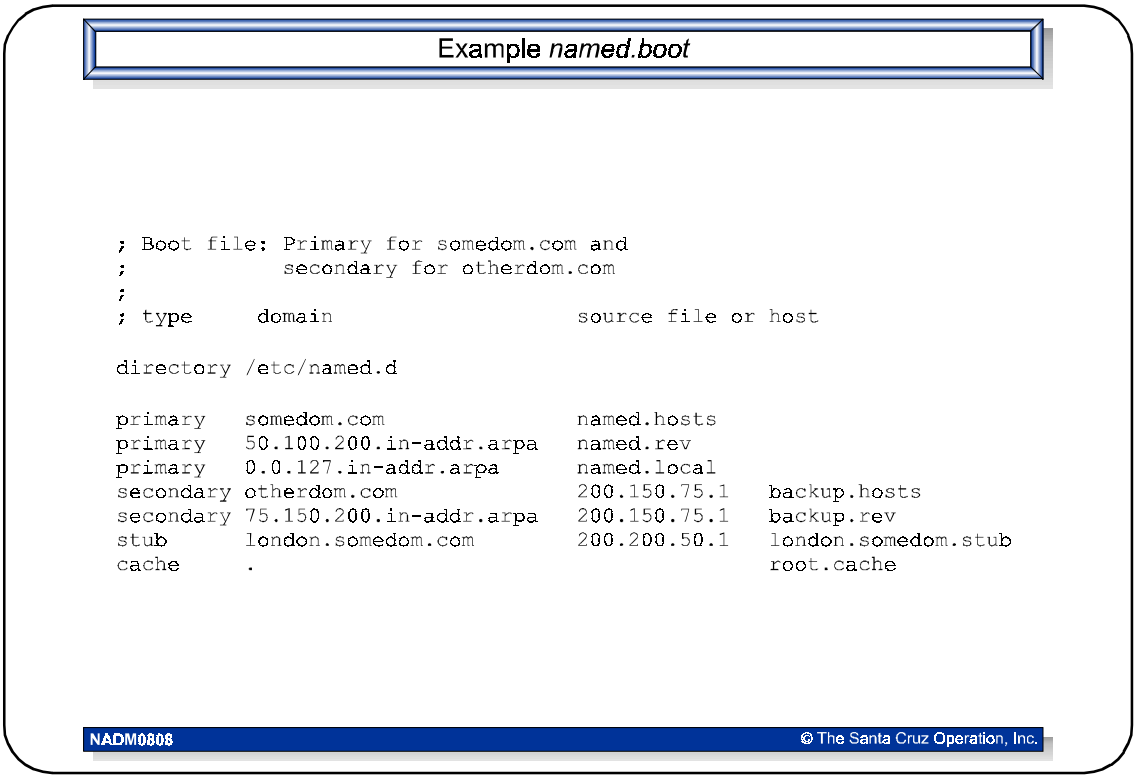
AV 8-7 shows the relationships between these DNS server configuration files.

The DNS configuration files (except *named.boot*) contain records 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. The next sections describe the format and contents of each configuration file.

DNS client configuration is described in Section 8.7 on page 8-67.

AV 8-7: File Relationships
Use this slide throughout the discussion of the files to help students keep perspective during the editing of the files. Although the file names are configurable, the “standard” files are used in this lesson to help simplify this complex subject. The *named.soa* file, which is often used when setting the files up by hand, is also used in this lesson because all DNS configuration is manual on SCO OpenServer Release 5.

AV 8-8


Example named.boot
; Boot file: Primary for somedom.com and
;   secondary for otherdom.com
;
; type   domain   source file or host
directory /etc/named.d
primary   somedom.com   named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa   named.local
secondary otherdom.com   200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub   london.somedom.com   200.200.50.1   london.somedom.stub
cache   .   root.cache
NADM0808
© The Santa Cruz Operation, Inc.

8.2.1 The *named.boot* file

AV 8-8: Example *named.boot*
This slide shows an example *named.boot* file for a fictitious domain. It is repeated throughout this lesson to help illustrate the components of DNS.

When **named**(ADMN) is started, it first looks for the */etc/named.boot* file. AV 8-8 shows an example of a *named.boot* file with these attributes:

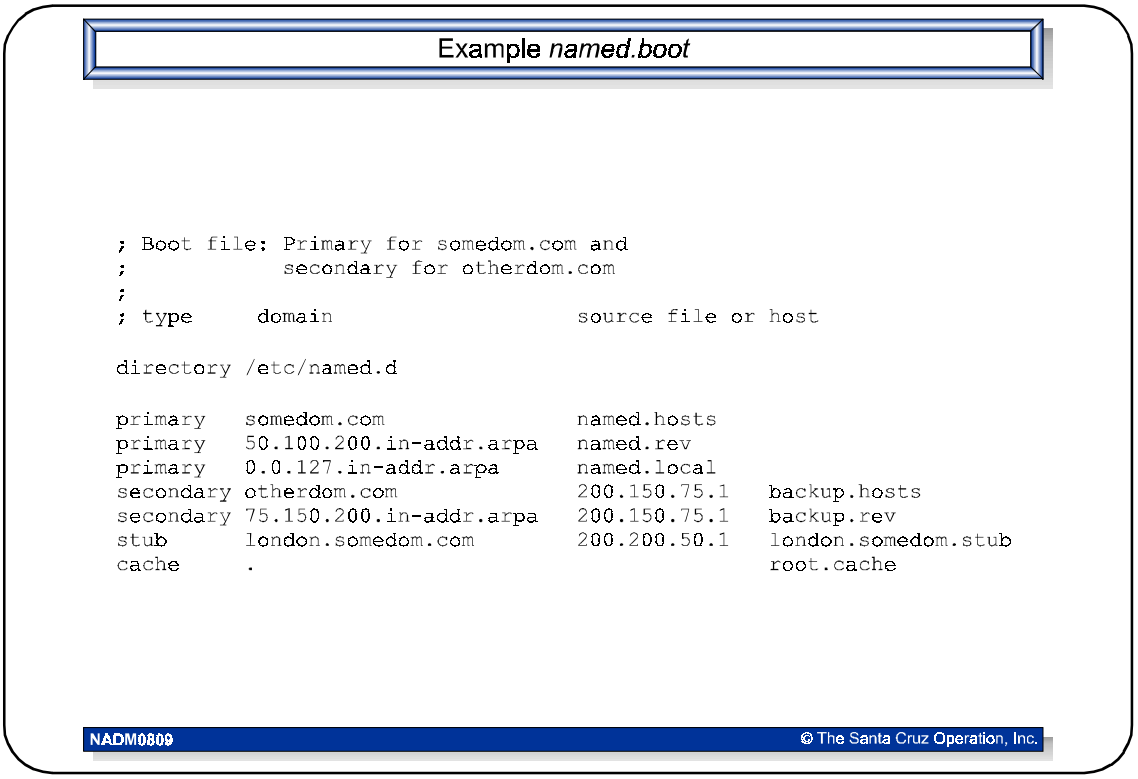
- It acts as a primary name server for the *somedom.com* domain.
- It acts as a secondary name server for the *otherdom.com* domain.
- The *somedom.com* domain has a network IP address of 200.100.50.
- The *otherdom.com* domain has a network IP address of 200.150.75.
- It acts as the stub server for the subdomain *london.somedom.com*.

See the Section , “Instructor preparation,” on page 8-3 for a discussion of the new **named** configuration file for BIND 8.1.1, *named.conf*, and how it relates to *named.boot*.

SCO OpenServer Release 5.0.5 includes an updated version of BIND (8.1.1) which uses *named.conf* as the primary **named** configuration file. Full backward compatibility is maintained with the *named.boot* file, however when **named** starts, the *named.boot* file is converted automatically to *named.conf*. Utilities are also provided to manually convert a *named.boot* file to *named.conf*. See *named.conf*(SFF) for more information.

This lesson covers *named.boot* configuration, because it is fully compatible with all releases of SCO OpenServer. It is also used in the BIND implementations found on many other UNIX systems.

AV 8-9


Example named.boot
; Boot file: Primary for somedom.com and
;   secondary for otherdom.com
;
; type   domain   source file or host
directory /etc/named.d
primary   somedom.com   named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa   named.local
secondary otherdom.com   200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub   london.somedom.com   200.200.50.1   london.somedom.stub
cache   .   root.cache
NADM0809
© The Santa Cruz Operation, Inc.

AV 8-9: Example *named.boot*
This slide repeats the example *named.boot* file to help describe DNS records and files.

8.2.1.1 The `directory` entry

The `directory` entry tells **named** where to find files referenced by *named.boot*. It is usually set to:

- */etc/named.d* (SCO OpenServer Release 5)
- */etc/inet/named.d* (UnixWare 7)
- */usr/lib/named* (previous releases of SCO UNIX)

8.2.1.2 The `primary` entry

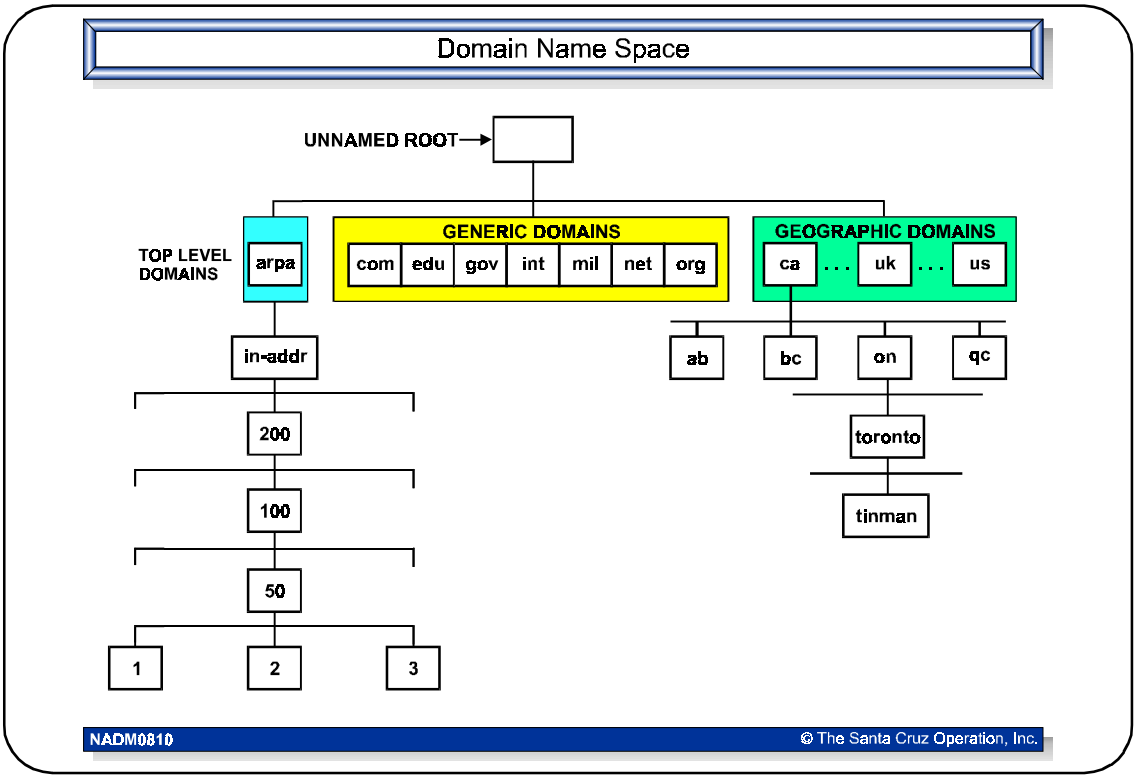
The `primary` entry tells **named** that:

- It is the primary name server for the zone indicated in the next field.
- The data files for hosts in this zone are named in the last field.

There are three entries for a primary name server:

- The first resolves queries by name.
- The second resolves queries by IP address (discussed in the next section).
- The third is for the *localhost*.

AV 8-10


Domain Name Space
UNNAMED ROOT
TOP LEVEL DOMAINS
arpa
GENERIC DOMAINS
com
edu
gov
int
mil
net
org
GEOGRAPHIC DOMAINS
ca
uk
us
in-addr
200
100
50
1
2
3
ab
bc
on
qc
toronto
tinman
NADM0810
© The Santa Cruz Operation, Inc.

8.2.1.2.1 The in-addr.arpa domain

The *in-addr.arpa* domain is organized by IP address, not domain name. It allows clients to look up 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.

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 a client to quickly look up an IP addresses by machine name.

- 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.

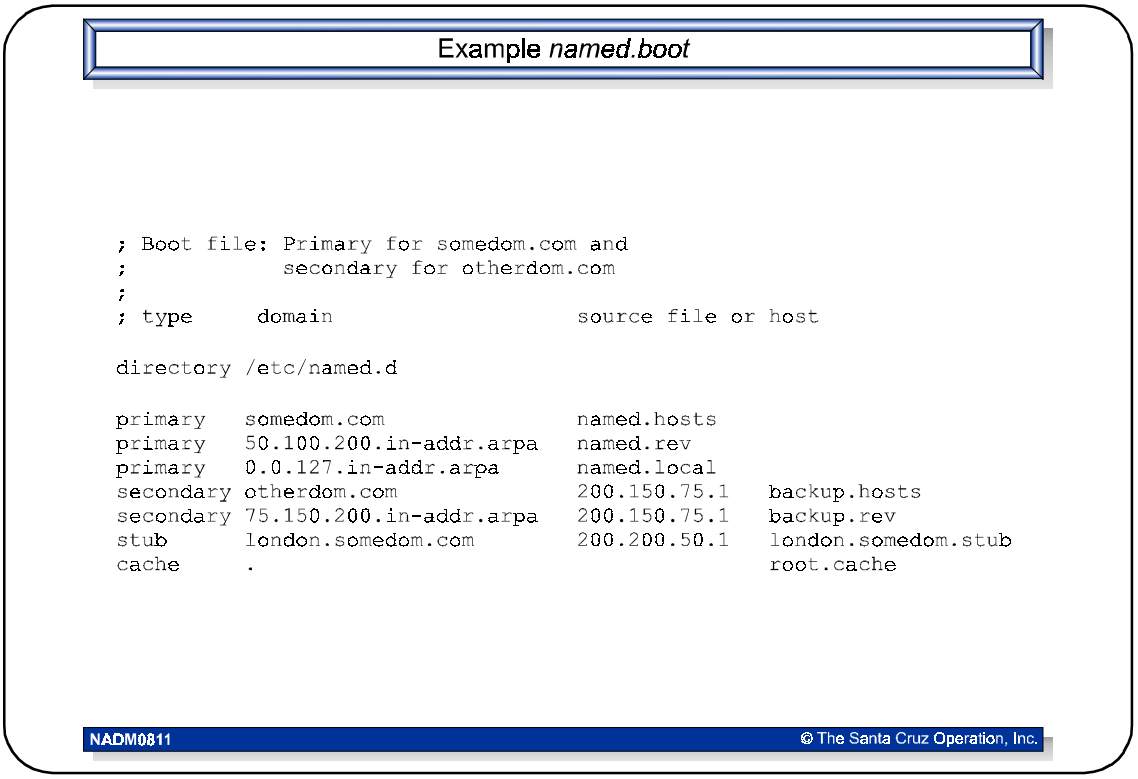
The *in-addr.arpa* domain solves this problem by writing the IP addresses in reverse order. AV 8-10 shows entries in the *in-addr.arpa* domain for the first few addresses in the *somedom.com* domain.

AV 8-10: Domain Name Space
Use this slide to illustrate why the *in-addr.arpa* domain is needed and how it works to resolve IP addresses into host names. It is similar to the slide in AV 2-9 on page 2-36.

You might have to spend some time discussing why reverse address entries are needed, and the existence of the *in-addr.arpa* domain.

The domain tree does not keep IP addresses in standard order. The *in-addr.arpa* domain was invented to keep all the IP addresses for a given network number under the same domain tree (that is, serviceable by a single name server).

AV 8-11


Example named.boot
; Boot file: Primary for somedom.com and
;   secondary for otherdom.com
;
; type   domain   source file or host
directory /etc/named.d
primary   somedom.com   named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa   named.local
secondary otherdom.com   200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub   london.somedom.com   200.200.50.1   london.somedom.stub
cache   .   root.cache
NADM0811
© The Santa Cruz Operation, Inc.

8.2.1.3 The `secondary` entry

AV 8-11: Example *named.boot*
This slide repeats the example *named.boot* file to help describe DNS records and files.

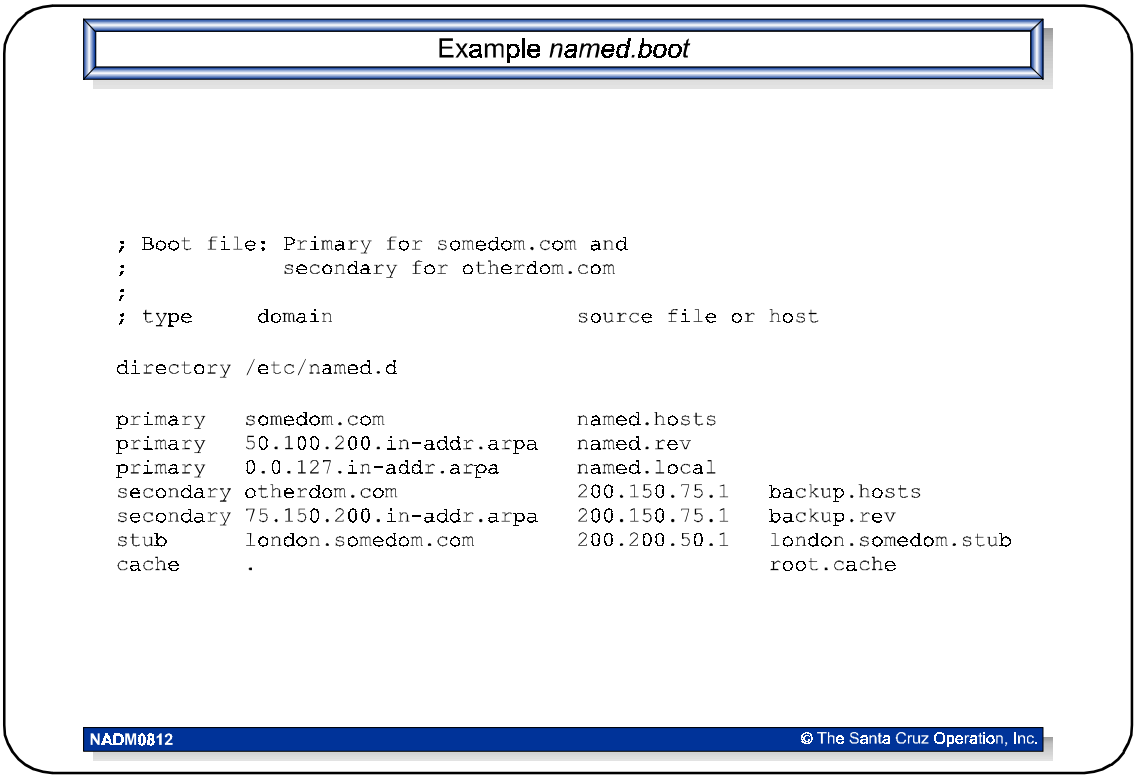
The `secondary` entry tells **named:**

- It is the secondary name server for the zone indicated in the second field.
- The IP address of the primary server for the zone is in the third field.
- Optionally, the data for that zone is stored in the file specified in the last field. If a filename is:

Specified	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** on the SNS can answer queries. Remember that the data file is only a backup. The primary server always has the original data.

NOTE: As with primary name servers, each zone must have two secondary name server entries: one to resolve lookups by name, and one to resolve lookups by IP address.

AV 8-12


Example named.boot
; Boot file: Primary for somedom.com and
;   secondary for otherdom.com
;
; type   domain   source file or host
directory /etc/named.d
primary   somedom.com   named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa   named.local
secondary otherdom.com   200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub   london.somedom.com   200.200.50.1   london.somedom.stub
cache   .   root.cache
NADM0812
© The Santa Cruz Operation, Inc.

8.2.1.4 The `forwarders` entry

AV 8-12: Example *named.boot*
This slide repeats the example *named.boot* file to help describe DNS records and files.

The `forwarders` entry is not included in the example *named.boot* file. When present, it:

- Tells **named** to forward requests when it does not know the answer to one or more other name servers
- Lists the IP addresses for each server to forward requests to, separated by white space
- 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 name server cache

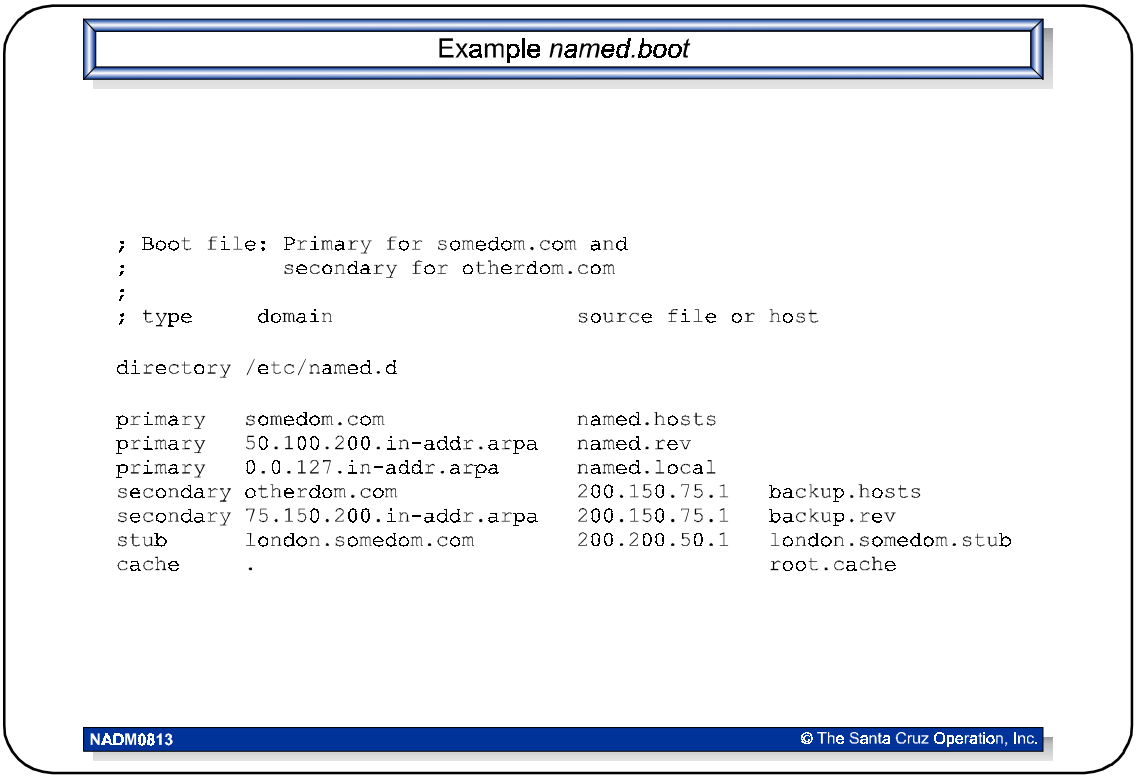
Forwarders send recursive queries as clients do instead of performing iterative queries like standard name servers. After a short period of time, forwarders resort to normal server operation if resolution has not succeeded.

An example is a configuration using a forwarder whose IP address is 200.100.50.1. The *named.boot* file contains this entry anywhere after the *directory* entry:

```
forwarders 200.100.50.1
```

Forwarders are often used in a firewall situation when the client does not have full Internet access. The requests are sent to the forwarder which is on the protected side of the firewall, but can send packets through the firewall and can therefore query other name servers on the Internet. Another reason to use a forwarder is that the forwarder, seeing all queries that pass through it, will build up a substantial cache of data, making answers to queries on that data much quicker.

AV 8-13


Example named.boot
; Boot file: Primary for somedom.com and
;   secondary for otherdom.com
;
; type   domain   source file or host
directory /etc/named.d
primary   somedom.com   named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa   named.local
secondary otherdom.com   200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub   london.somedom.com   200.200.50.1   london.somedom.stub
cache   .   root.cache
NADM0813
© The Santa Cruz Operation, Inc.

AV 8-13: Example *named.boot*
This slide repeats the example *named.boot* file to help describe DNS records and files.

8.2.1.5 The `slave` entry

The `slave` entry:

- 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, such as when 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.

The `slave` entry is added to *named.boot* after the `directory` entry.

This can also be implemented using the `forward-only` option in the *named.conf* file.

8.2.1.6 The `cache` entry

The `cache` entry tells **named** to:

- Initialize its cache with data for the domain named in the second field
- Get cache data from the file specified in the last field

The domain specified in the second field is almost always the *root* domain (`.`), with data found in the *root.cache* file.

AV 8-14

Example *named.boot*

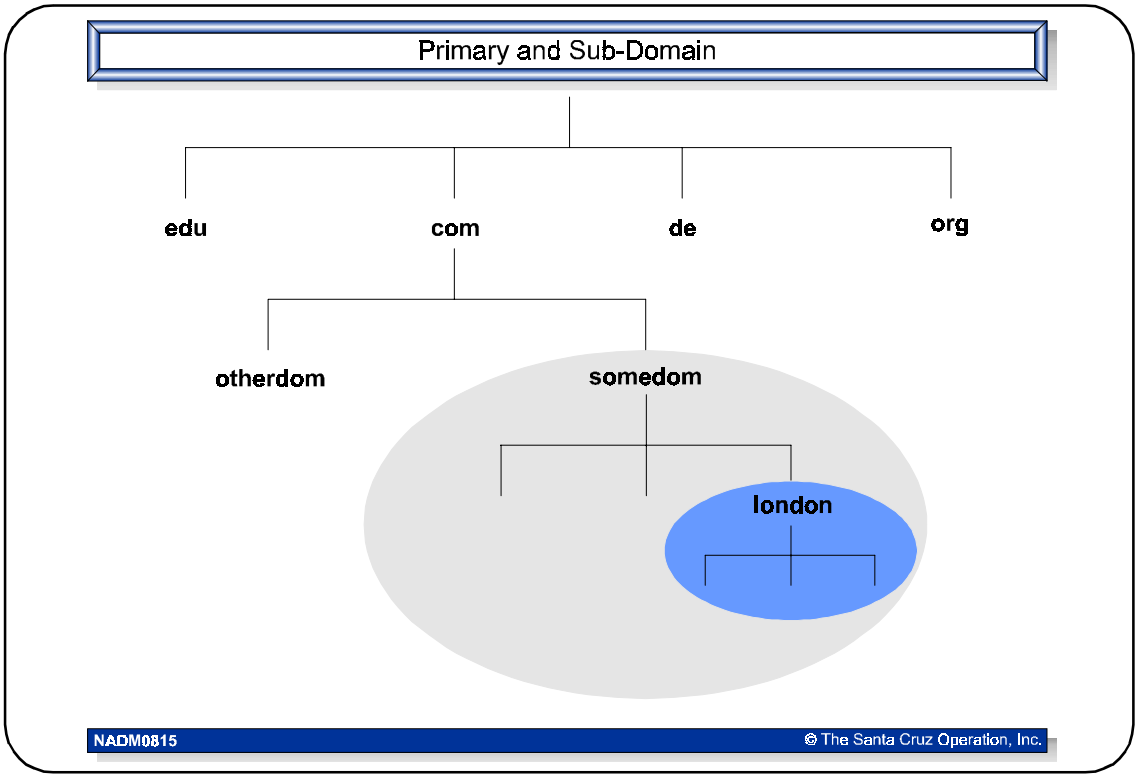
```
; Boot file: Primary for somedom.com and
;            secondary for otherdom.com
;
; type     domain                    source file or host

directory /etc/named.d

primary   somedom.com               named.hosts
primary   50.100.200.in-addr.arpa   named.rev
primary   0.0.127.in-addr.arpa      named.local
secondary otherdom.com              200.150.75.1   backup.hosts
secondary 75.150.200.in-addr.arpa   200.150.75.1   backup.rev
stub      london.somedom.com        200.200.50.1   london.somedom.stub
cache     .                                        root.cache
```

NADM0814 © The Santa Cruz Operation, Inc.

AV 8-15



8.2.1.7 The `stub` entry

AV 8-14: Example *named.boot*
This slide repeats the example *named.boot* file to help describe DNS records and files.

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 second field.
- The IP address of the primary server for the sub-domain is in the third field.
- Optionally, the data for the sub-domain is stored in the file specified in the last field. If a filename is:
  - Specified, then the name server loads the data from disk. Then it contacts the primary to make sure the data is up to date by comparing 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 data.

The data referred to here is NOT the same as for an SNS.The stub directive instructs **named** to obtain data regarding NS entries from the specified host. In contrast, an SNS transfers all the records for the zone.

In AV 8-14, 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 8-15: Primary and Sub-Domain
In this slide the *somedom.com* domain has a *london* sub-domain which is not covered by the name server as shown above. Rather, it has its own name server for that portion of the *somedom.com* domain.

AV 8-16

Example *named.hosts*

```
; 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
```

NADM0816 © The Santa Cruz Operation, Inc.

AV 8-17

Example *named.rev*

```
; name [ttl] class type data

$include named.soa

1               IN    PTR   thatch.somedom.com.

2               IN    PTR   guam.somedom.com.

3               IN    PTR   atlantis.somedom.com.

4               IN    PTR   gateway.somedom.com.
```

NADM0817 © The Santa Cruz Operation, Inc.

The remaining configuration files are discussed below. Each describes the records typically included in the file. It is not required that the file names follow this convention. Nor is it necessary to strictly follow the convention of including certain records within certain files. The descriptions in this guide are meant to be consistent with generally established convention, as well as the documentation, to try to make this as easy to learn as possible. Rather than describing all configuration possibilities, this lesson tries to focus on a basic, working configuration.

AV 8-16: Example *named.hosts*
This slide shows an example *named.hosts* file.

8.2.2 The *named.hosts* file

The *named.hosts* file, shown in AV 8-14, maps host names to IP addresses. It typically includes:

- The *named.soa* file by reference
- One A record for each host in the zone
- CNAME records for creating aliases or nicknames for hosts

See Section 8.3.1, “The A record,” on page 8-47 and Section 8.3.2, “The CNAME record,” on page 8-47 for details on these records.

8.2.3 The *named.rev* file

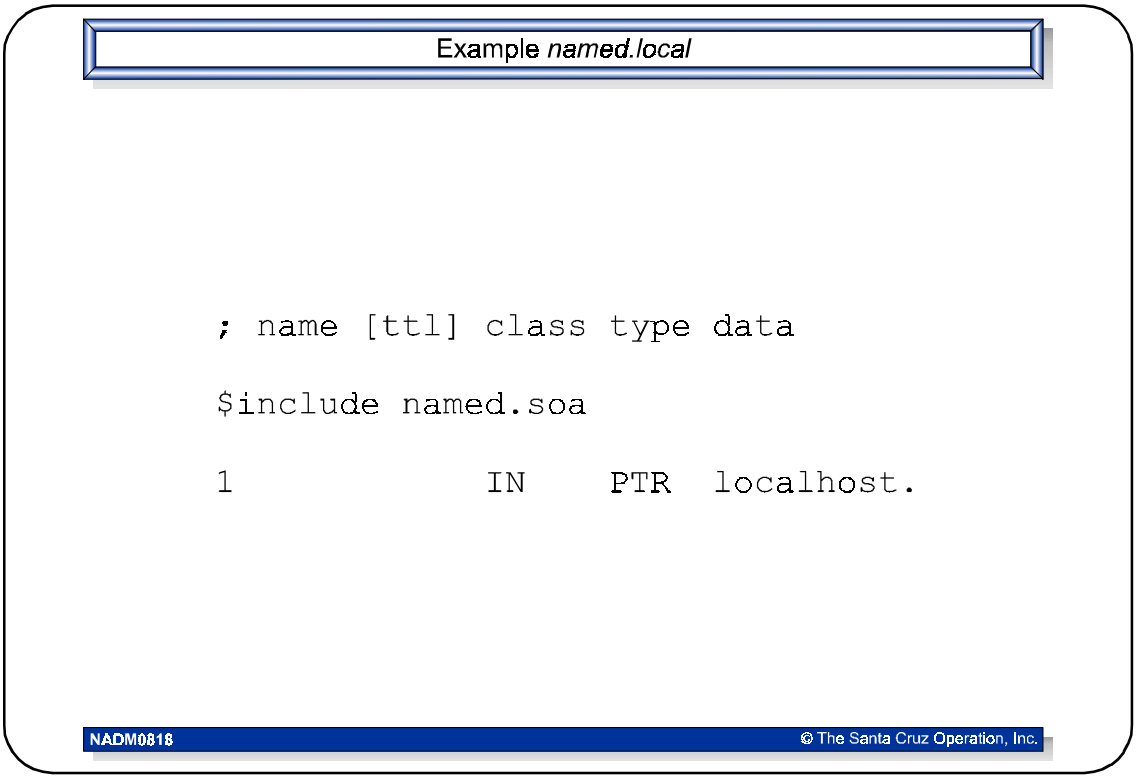
The *named.rev* file, shown in AV 8-17, maps IP addresses to host names. It includes:

- The *named.soa* file by reference
- One PTR record for each host in the zone

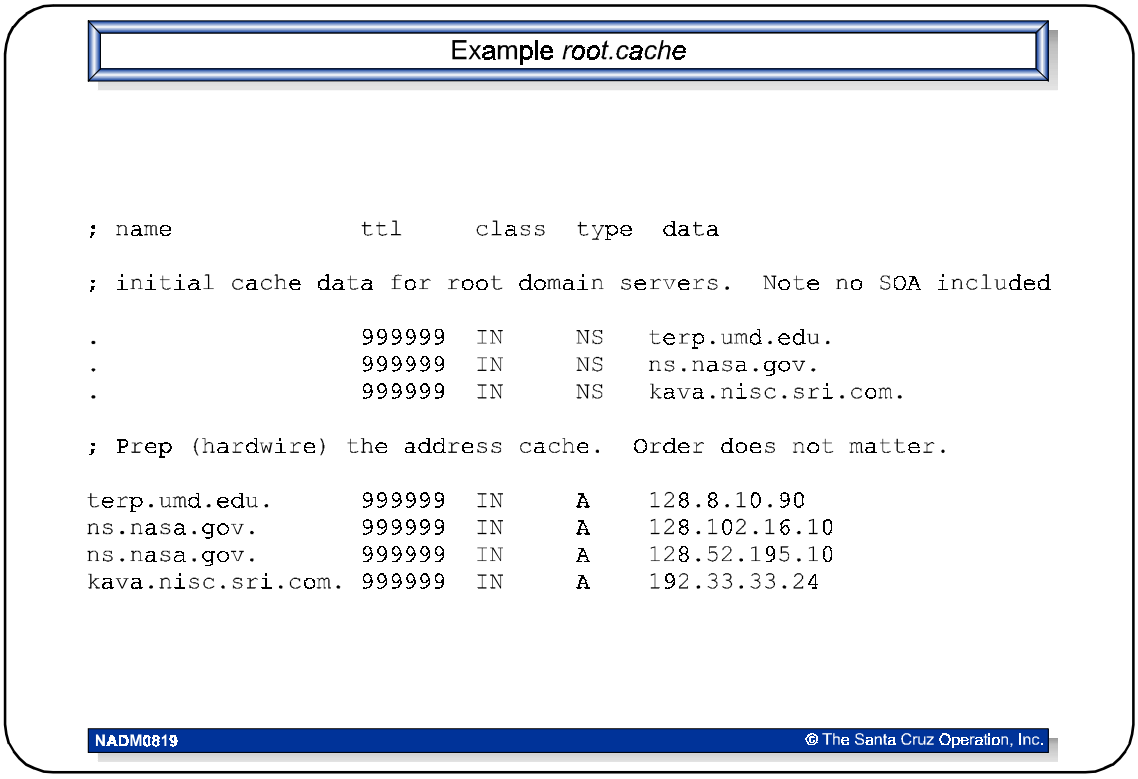
See Section 8.3.3, “The PTR Record,” on page 8-51 for details on the format of PTR records.

AV 8-17: Example *named.rev*
This slide shows an example *named.rev* file.

AV 8-18


Example named.local
; name [ttl] class type data
$include named.soa
1 IN PTR localhost.
NADM0818
© The Santa Cruz Operation, Inc.

AV 8-19


Example root.cache
; 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
NADM0819
© The Santa Cruz Operation, Inc.

AV 8-18: Example *named.local*
This slide shows an example *named.local* file.

8.2.4 The *named.local* file

The *named.local* file, shown in AV 8-18, translates IP address 127.0.0.1 to *localhost*. It includes:

- The *named.soa* file by reference
- One PTR record for *localhost*

See Section 8.3.3, “The PTR Record,” on page 8-51 for details on the format of PTR records.

8.2.5 The *root.cache* file

The *root.cache* file, shown in AV 8-14, contains information about the name servers for the root domain. It contains:

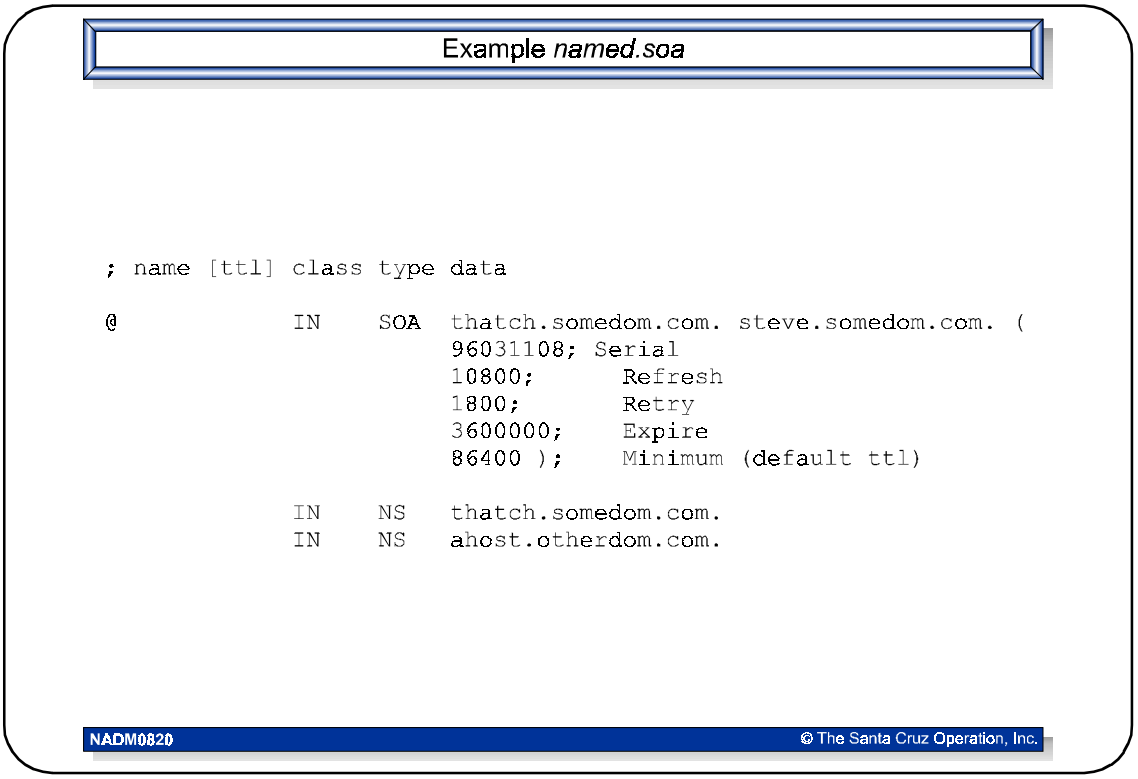
- A records for all of the root name servers
- NS records indicating that each of these machines are name servers for the root domain (.)

These records are described in more detail in Section 8.3.1, “The A record,” on page 8-47 and Section 8.3.4, “The NS record,” on page 8-53.

The *root.cache* file should be updated periodically to the current version. It is available using anonymous **ftp** from *ftp.rs.internic.net*.

AV 8-19: Example *root.cache*
This slide illustrates the contents of the *root.cache* file. Use this slide to also illustrate the format of the NS records. NOTE: Remind students that they should get an updated copy initially and once every month or two thereafter. Also, point out the need for the A records.

AV 8-20


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.
NADM0820
© The Santa Cruz Operation, Inc.

AV 8-20: Example *named.soa*
This slide shows an example *named.soa* file.

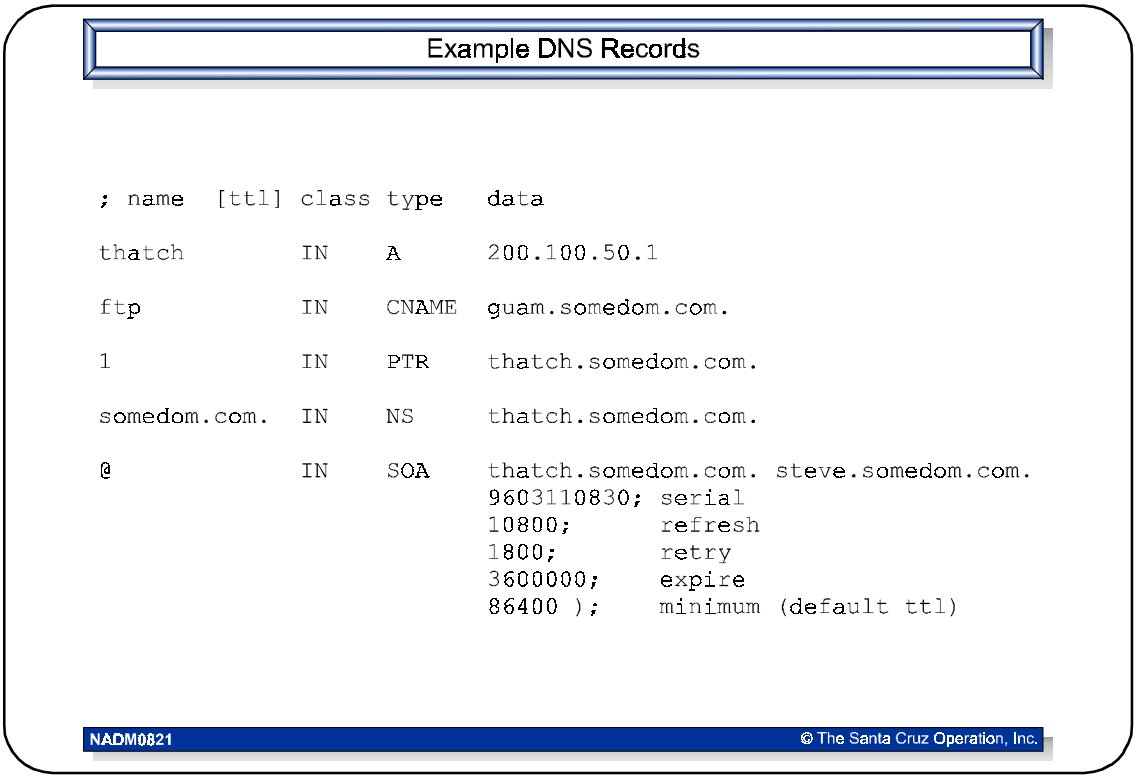
8.2.6 The *named.soa* file

The *named.soa* file, shown in AV 8-14, contains the *start of authority* record for one zone.

- It is included by reference in *named.hosts*, *named.rev,* and *named.local*.
- It contains:
  - One `SOA` record
  - One or more `NS` records listing name servers in the domain

These records are described in more detail in Section 8.3.5, “The `SOA` record and `$include`,” on page 8-55 and Section 8.3.4, “The `NS` record,” on page 8-53.

AV 8-21


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)
NADM0821
© The Santa Cruz Operation, Inc.

8.3 DNS Records

AV 8-21: Example DNS Records
This slide shows several example records in SRR format. Use these examples as you step through the fields below. Have students bring up the *named.hosts*(SFF) manual page and describe each field using this slide.

Primary name servers read the information for their zone from the DNS configuration files, which contain data in the SRR-format. AV 8-17 shows examples of various records in the SRR format.

Characters with special meaning in DNS records

Some characters that appear in DNS records have special meaning. Rather than overwhelming students with this level of detail before they see example records, it is included as a reminder for the instructor to use while explaining the example records.

- A free-standing at-sign (@) in the "name" field denotes the name of the current zone as indicated by the second field in *named.boot*.

  For example, if *named.boot* has: `primary xyz.com named.hosts`, and the @ appears in *named.hosts*, then @ would expand to `xyz.com`.

- A free-standing period (.) in the "name" field refers to the current domain. A period at the end of a host name indicates that the current zone name (the value of @) should not be appended to it. A host name that does not end in a period will have the current zone name (the value of @) appended to it.

  For example, if the machine is in the *sco.com* domain, then `.` would always refer to sco.com, regardless of which zone *named.boot* contained data for. For example, if the current zone name is *somedom.com*:

  - *foo* becomes *foo.somedom.com*
  - *foo.somedom.com* becomes *foo.somedom.com.somedom.com*
  - *foo.somedom.com.* remains as is

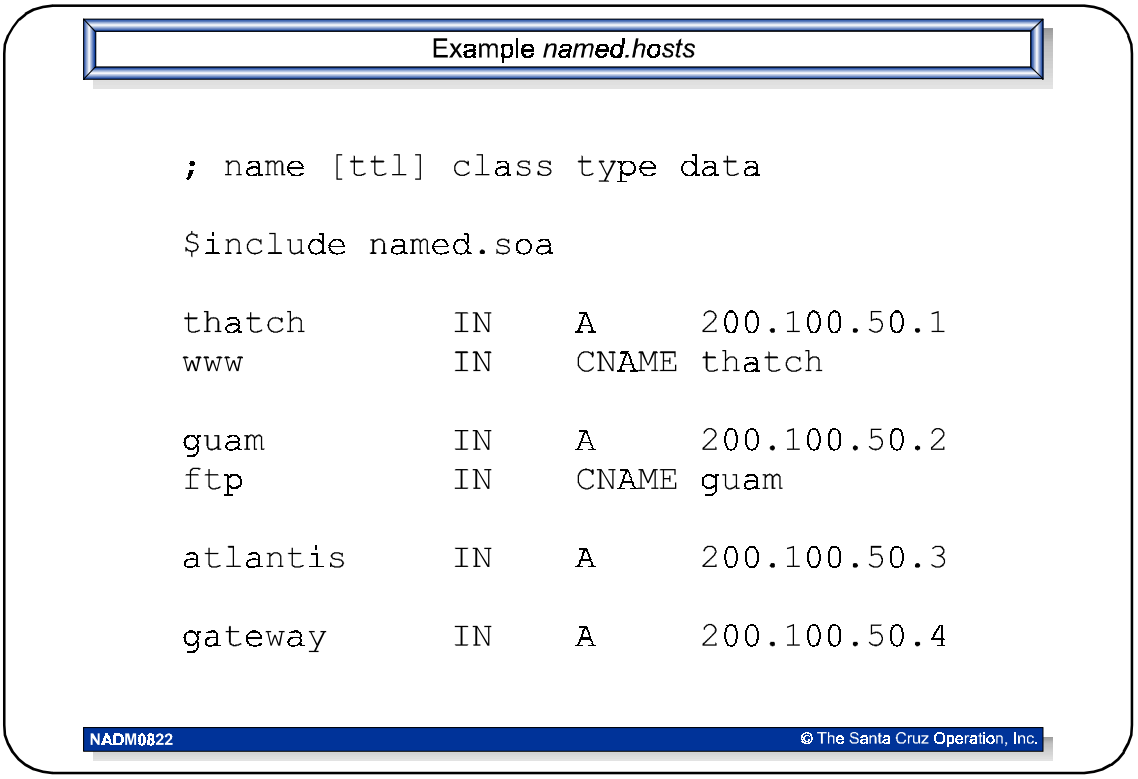
NOTE: The meaning of `".",` the *null* domain of the *root* domain, was left off because it is rarely used.

Other special characters include:

Empty space	Repeats the entry from that column in the line above
Semicolon (;)	Starts a comment that extends to the end of the line
Parentheses ( )	Group data for a record that spans across multiple lines

Asterisk (*)	Signifies a wildcard
Backslash (\)	Quotes the one non-digit character that follows it so that its special meaning does not apply in this one instance

AV 8-22


Example named.hosts
; 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
NADM0822
© The Santa Cruz Operation, Inc.

AV 8-22: Example *named.hosts*
This slide repeats the example *named.hosts* file to illustrate use of A and CNAME records.

8.3.1 The A record

A (address) records are used in *named.hosts* to map IP addresses to host names using these fields:

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 *localhost*.

You might need to explain again that it is okay to place *localhost* (by name) in the zone's data file, even though you cannot place it in the same data file as the reverse entries for the zone.

8.3.2 The CNAME record

CNAME (canonical name) records allow you to create aliases or nicknames for your hosts. They 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** can have problems with multiple names.

CNAME records use these fields:

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

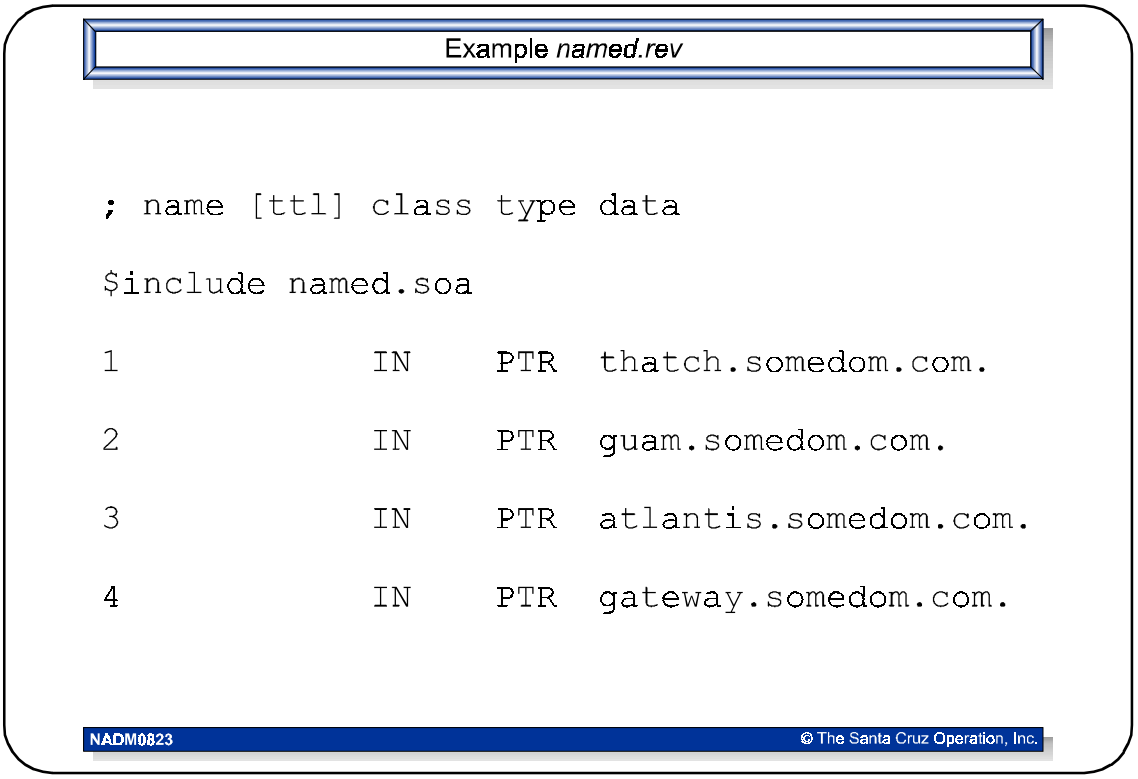
As well as having multiple names for one IP address, it might be useful to configure multiple IP address for one name. This can be done by entering multiple A records for each IP address. It is useful when you have a number of servers fulfilling the same function.

The name server, when queried for this name, returns all the applicable addresses in the order determined by the name server.

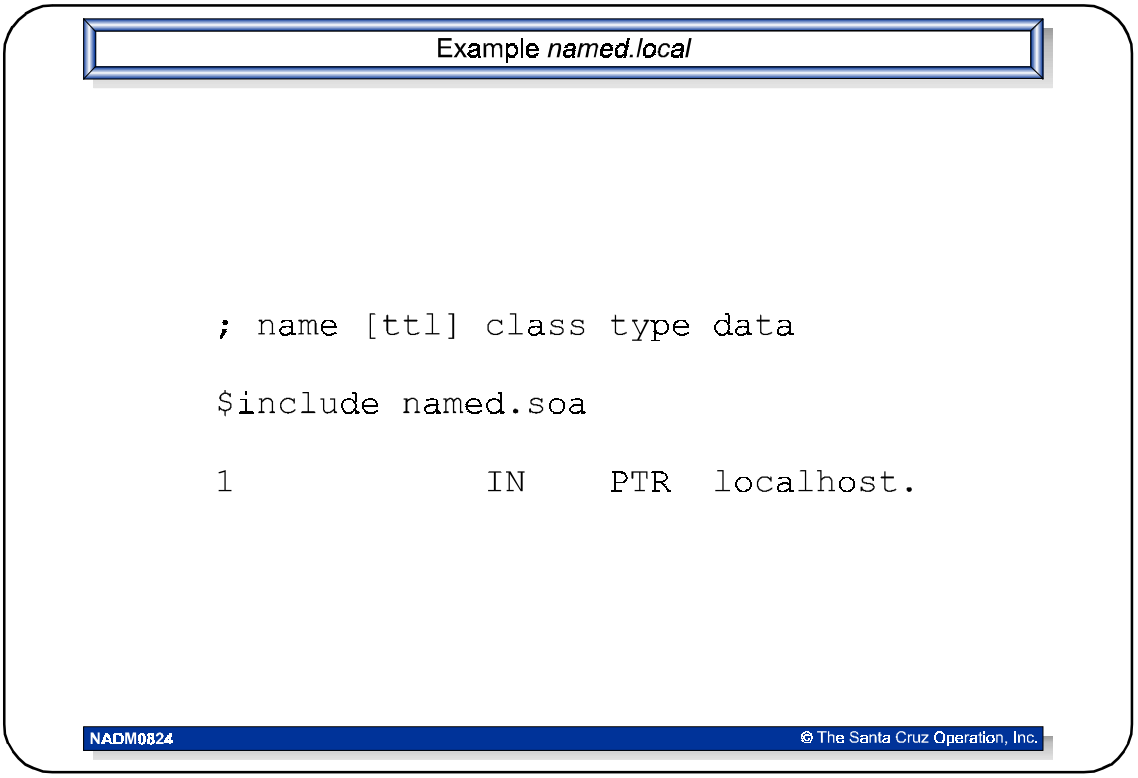
Give an example of the host name *www* leading to multiple IP addresses to distribute the load on busy web servers.

Some students might want to talk about MX and other such records at this point. Defer them to Lesson 12: "Mail Configuration".

AV 8-23


Example named.rev
; name [ttl] class type data
$include named.soa
1 IN PTR thatch.somedom.com.
2 IN PTR guam.somedom.com.
3 IN PTR atlantis.somedom.com.
4 IN PTR gateway.somedom.com.
NADM0823
© The Santa Cruz Operation, Inc.

AV 8-24


Example named.local
; name [ttl] class type data
$include named.soa
1 IN PTR localhost.
NADM0824
© The Santa Cruz Operation, Inc.

8.3.3 The PTR Record

AV 8-23: Example *named.rev*
This slide illustrates the use of PTR records in the reverse look up file.

If the netmask extends into the host portion, does each subnet need a separate reverse-address file? No. All hosts that have some part of their IP address in common can be listed in the same file, as long as the common portion of the IP address is listed in the *named.boot* file.

PTR (pointer) records map IP addresses to a host names. They are:

- Found in the *in-addr.arpa* domain files such as *named.rev*
- Typically the only records included other than an SOA record

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.boot* file.

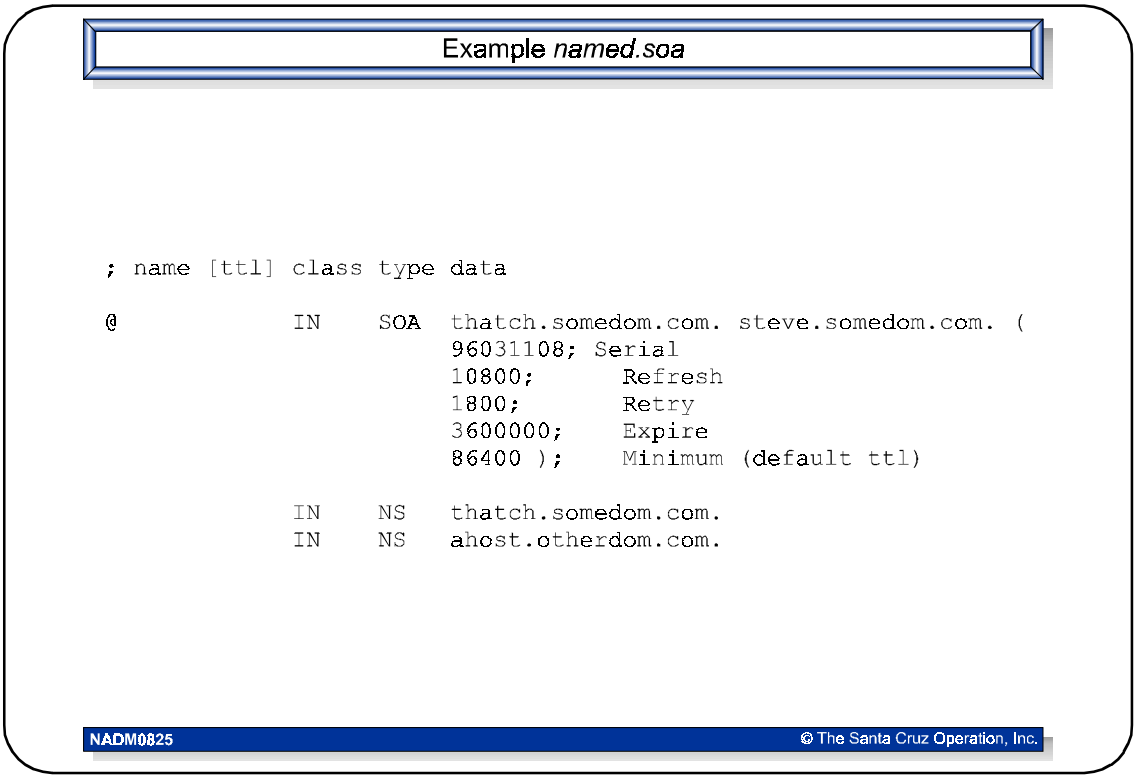
PTR records use these fields:

name	Contains 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 *named.local* file includes an SOA and a single PTR record for the local host. AV 8-24 illustrates a *named.local* file.

AV 8-24: Example *named.local*
This slide illustrates the way that all *named.local* files should look. Point out that the reason that this works the same for all machines is that the IP address 127.0.0.1 is special and is *localhost* on all hosts.

AV 8-25


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.
NADM0825
© The Santa Cruz Operation, Inc.

8.3.4 The NS record

NS records provide information about which hosts are providing names services for specified domains. NS records contain these fields:

name	Contains 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 *root.cache* 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 *named.soa* file. The domain name could be blank, indicating that this record should use the same domain name as the previous record.
time-to-live	Left blank in the *named.soa* file entries. Set to 999999 in *root.cache* 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 8-25: Example *named.soa*
This slide repeats the example *named.soa* file to illustrate use of NS records.

AV 8-26

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.
```

NADM0826 © The Santa Cruz Operation, Inc.

AV 8-27

Example *named.hosts*

```
; 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
```

NADM0827 © The Santa Cruz Operation, Inc.

AV 8-26: Example *named.soa*
This slide illustrates the use of the SOA record file. Point out the periods after the fully-qualified host names in the first line of the SOA record to reinforce the use of trailing periods.

8.3.5 The SOA record and $include

Several references have been made to the *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.

Review the relationship between the primary and secondary name servers.

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.
- 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 shown in AV 8-27. This *named.hosts* file references the *named.soa* file displayed in AV 8-26.

Explain how direct inclusion can be more efficient because only data files that change need to be retransmitted to secondary servers. The indirect method can be easier to maintain than configuring files directly.

AV 8-27: Example *named.hosts*
This slide shows how *named.soa* is included in *named.hosts*.

These SOA record includes these fields:

name	Set to the domain name
address class	Set to IN for Internet
record type	Set to SOA
host name	The name of the authoritative host for this domain
email address	The email 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 serial number 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:

  - *YYYY* is the year of the change
  - *MM* is the month of the change
  - *DD* is the day of the change
  - *HH* is the hour of the change
  - *XX* is a counter for changes that hour

  This method also has the benefit of indicating when the last data was last changed.

8.4 Configuring DNS servers

After you have decided how to structure DNS services for your particular network, you will need to set up one primary, one or more secondary, and any number of caching-only name servers.

Note that you must also configure each host as a DNS client by creating an */etc/resolv.conf* file, including the hosts that are also name servers. See Section 8.7, “Configuring DNS Clients,” on page 8-67.

The basic procedure for configuring any server is to create the files described in Section 8.2, “DNS Files,” on page 8-21, according to the specifications for your network.

The steps to configure DNS on a simple example network are shown in the next sections. This example network has these attributes:

Domain name	*somedom.com*
Primary name server	*thatch*, 200.100.50.1
Secondary name server	*rosco*, 200.100.50.5
Caching-only name server	*cush*, 200.100.50.6
Hosts	*guam*, 200.100.50.2 *atlantis*, 200.100.50.3 *gateway*, 200.100.50.4

8.4.1 Configuring a primary name server

To configure the system, *thatch*, as a primary name server for the example network:

Step 1. Create /*etc/named.boot* with these entries:

```
directory   /etc/named.d
primary     somedom.com                 named.hosts
primary     50.100.200.in-addr.arpa     named.rev
primary     0.0.127.in-addr.arpa        named.local
cache       .                           root.cache
```

Step 2. Create */etc/named.d/named.hosts* with an A record for each host:

```
$INCLUDE    named.soa
localhost   IN A                    127.0.0.1
thatch      IN A                    200.100.50.1
guam        IN A                    200.100.50.2
atlantis    IN A                    200.100.50.3
gateway     IN A                    200.100.50.4
rosco       IN A                    200.100.50.5
cush        IN A                    200.100.50.6
```

Step 3. Create /*etc/named.d/named.rev* with a PTR record for each host:

```
$INCLUDE    named.soa
1           IN PTR          thatch.somedom.com.
2           IN PTR          guam.somedom.com.
3           IN PTR          atlantis.somedom.com.
4           IN PTR          gateway.somedom.com.
5           IN PTR          rosco.somedom.com.
6           IN PTR          cush.somedom.com.
```


Step 4. Create /*etc/named.d/named.local* with the following entry:

```
$INCLUDE    named.soa
1           IN PTR           localhost.
```

Step 5. If not already present, create /*etc/named.d/root.cache.* Copy this file from another server or get the latest version from *ftp.rs.internic.net.*

Step 6. Create /*etc/named.d/named.soa* with these entries:

```
@           IN   SOA  thatch.somedom.com. ned.somedom.com. (
                      199809180201;    Serial
                      10800;           Refresh
                      1800;            Retry
                      360000;          Expire
                      86400 );         Minimum (default ttl)
            IN   NS   thatch.somedom.com.
            IN   NS   rosco.somedom.com.
```

Step 7. Create */etc/resolv.conf* with these entries:

```
name server 200.100.50.1
hostresorder local bind
```

The */etc/resolv.conf* file is discussed in more detail in Section 8.7.1 on page 8-67.

Step 8. Start **named** as described in Section 8.5 on page 8-61.

8.4.2 Configuring a secondary name server

To configure the system, *rosco*, as a secondary name server for *somedom.com*:

Step 1. Create /*etc/named.boot* with these entries:

```
directory  /etc/named.d
secondary  somedom.com              200.100.50.1 somedom.hosts
secondary  50.100.200.in-addr.arpa 200.100.50.1 somedom.rev
primary    0.0.127.in-addr.arpa    named.local
cache      .                       root.cache
```

Note the files *somedom.hosts* and *somedom.rev* are created when **named** is started. You do not create these manually.

Step 2. Create /*etc/named.d/named.local* with this entry:

```
$INCLUDE    named.soa
1           IN PTR                    localhost.
```

Step 3. If not already present, create /*etc/named.d/root.cache.* Copy this file from another server, or get the latest version from *ftp.rs.internic.net*.

Step 4. Create /*etc/named/named.soa* with these entries:

```
@           IN   SOA  rosco.somedom.com. ned.somedom.com. (
                      980918;    Serial
                      10800;     Refresh
                      1800;      Retry
                      360000;    Expire
                      86400 );   Minimum (default ttl)
            IN NS     rosco.somedom.com.
```

Step 5. Create */etc/resolv.conf* with these entries:

```
name server 200.100.50.5
name server 200.100.50.1
name server 200.100.50.6
hostresorder local bind
```

The */etc/resolv.conf* file is discussed in more detail in Section 8.7.1, “The /etc/resolv.conf file,” on page 8-67.

Step 6. Start **named** as described in Section 8.5 on page 8-61.

8.4.3 Configuring a caching-only name server

To configure *cush* as a caching-only server on a system:

Step 1. Create */etc/named.boot* with these entries:

```
directory  /etc/named.d
cache      .                          root.cache
primary    0.0.127.in-addr.arpa       named.local
```

Step 2. Create /*etc/named.d/named.local* with this entry:

```
$INCLUDE    named.soa
1           IN PTR                    localhost.
```

Step 3. If not already present, create /*etc/named/root.cache*. Copy this file from another server, or get the latest version from *ftp.rs.internic.net*.

Step 4. Create /*etc/named.d/named.soa* with these entries:

```
@           IN   SOA  cush.somedom.com. ned.somedom.com. (
                      980918;    Serial
                      10800;     Refresh
                      1800;      Retry
                      360000;    Expire
                      86400 );   Minimum (default ttl)
            IN NS     cush.somedom.com.
```

Step 5. Create */etc/resolv.conf* with these entries:

```
name server 200.100.50.1
name server 200.100.50.5
name server 200.100.50.6
hostresorder local bind
```

Step 6. Start **named** as described in Section 8.5 on page 8-61.

8.5 Starting and Controlling the Name Server

After you have set up all of the necessary configuration files, start the name server by either rebooting the system or using the **named** control interface, **ndc**(ADMN) to start the **named** daemon. From the command line, enter:

```
ndc start
```

The **ndc**(ADMN) command was not available in early 5.0 releases. In pre-5.0.5 releases, **named** can safely be started from the command line. However, in 5.0.5, **named** will not check for a *named.boot* file and will fail to start if *named.conf* does not exist.

To signal **named** to log debugging information, use the **trace** option to **ndc**:

```
ndc trace
```

This places diagnostic information in the */etc/named.d/named.run* file. This diagnostic information can be very useful for troubleshooting configuration problems. Increasingly-detailed information is logged with successive **ndc trace** commands.

When **named** starts, it stores its process ID number in */etc/named.pid*. Signals can be sent to the name server to affect its operation using this PID number and the **kill**(C) command. For example, to signal the name server to reread its configuration files, enter:

```
kill -HUP ‘cat /etc/named.pid‘
```

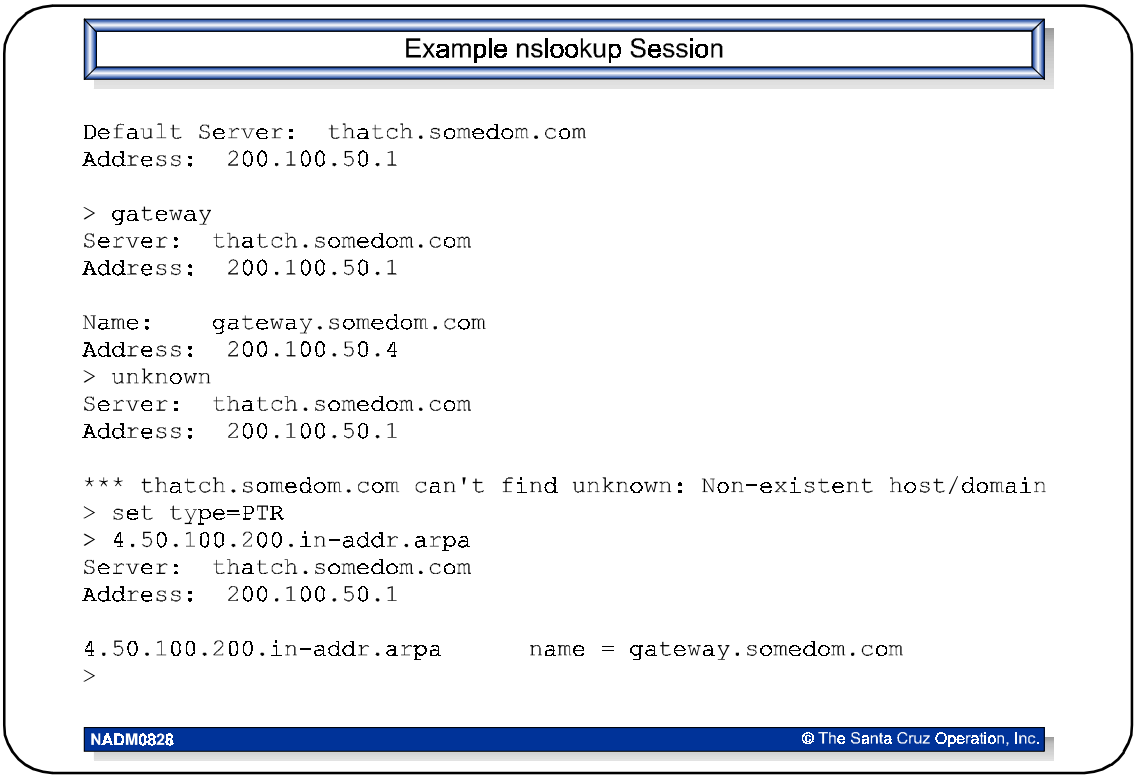
This can also be done with the command line:

```
ndc restart
```

This is useful if you make direct modifications of the configuration files and want the name server to immediately reflect the changes. For other signals you can send the name server, refer to the **named**(ADMN) and **ndc**(ADMN) manual pages.

This might be a good time to do the lab in Section 8.9.1, “Configuring servers,” on page 8-74. The two follow-on labs, Section 8.9.2, “Testing your server,” on page 8-77 and Section 8.9.3, “Configuring and testing a client,” on page 8-77, can be done as exercises for the next sections.

AV 8-28


Example nslookup Session
Default Server: thatch.somedom.com
Address: 200.100.50.1
> gateway
Server: thatch.somedom.com
Address: 200.100.50.1
Name: gateway.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 = gateway.somedom.com
>
NADM0828
© The Santa Cruz Operation, Inc.

8.6 Querying a name server

After you have set up a name server, verify that it works using the utilities, **nslookup**(ADMN), **host**(ADMN), and **dig**(ADMN).

8.6.1 The nslookup(ADMN) command

The **nslookup** command is used in two ways:

1. Enter a series of **nslookup** commands 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, to start **nslookup** in interactive mode, enter:

   ```
   nslookup
   ```

   You should receive output similar to that shown in AV 8-28. You can now issue 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 multiscreen) and running **nslookup** in another. You can exit **nslookup** by entering the end-of-file character, CTRL D.

AV 8-28: Example nslookup Session
Use this slide to illustrate what students should expect to see when querying with **nslookup**.

Have students look up **nslookup**(ADMN) in the manual pages. Discuss some of the command-line options and interactive commands. Have them run **nslookup** in the various modes.

NOTE: You can only have students perform **nslookup** if they have already done the first part of the lab as suggested.

AV 8-29

The **dig**(ADMN) Command

```
dig -x 200.100.50.1
; <<>> DiG 8.1 <<>> -x
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1,ADDITIONAL: 1
;; QUERY SECTION:
;;  1.50.100.200.in-addr.arpa, type = ANY, class = IN

;; ANSWER SECTION:
1.50.100.200.in-addr.arpa.  1D IN PTR  thatch.somedom.com.

;; AUTHORITY SECTION:
50.100.200.in-addr.arpa.  1D IN NS  thatch.somedom.com.

;; ADDITIONAL SECTION:
thatch.somedom.com.    1D IN A    200.100.50.1

;; Total query time: 10 msec
;; FROM: thatch.somedom.com to SERVER: default -- 0.0.0.0
;; WHEN: Tue Sep 22 13:53:24 1998
;; MSG SIZE  sent: 46  rcvd: 132
```

NADM0829 © The Santa Cruz Operation, Inc.

AV 8-30

The **host**(ADMN) Command

```
host -a www.sco.com
Trying null domain
rcode = 0 (Success), ancount=1
The following answer is not authoritative:
www.sco.com   3578 IN   CNAME  www.external.sco.COM
For authoritative answers, see:
sco.COM    41978 IN  NS  SCONIGMA.sco.COM
sco.COM    41978 IN  NS  NS2.EXODUS.NET
Additional information:
SCONIGMA.sco.COM 41978 IN  A  132.147.210.10
NS2.EXODUS.NET   172698 IN A  207.82.198.150
rcode = 0 (Success), ancount=1
The following answer is not authoritative:
www.external.sco.COM    940 IN A  209.1.8.14
For authoritative answers, see:
external.sco.com 3578 IN   NS  dd-scca.sco.com
external.sco.com 3578 IN   NS  dd-jcnj.sco.com
external.sco.com 3578 IN   NS  dd-lnuk.sco.com
Additional information:
dd-scca.sco.com  3578 IN   A  207.82.198.237
dd-jcnj.sco.com  3578 IN   A  209.67.27.230
```

NADM0830 © The Santa Cruz Operation, Inc.

8.6.2 The dig(ADMN) command

Another useful tool, especially when trying to test the PTR entries of a name server is the Domain Information Groper command, **dig**(ADMN). To test a PTR record using **dig,** enter:

```
dig -x 200.100.50.1
```

Compare the output of **dig** in AV 8-32 with the **nslookup** command line:

```
nslookup 200.100.50.1
Server:  thatch.somedom.com
Address: 100.200.50.1

1.50.100.200.in-addr.arpa  name = thatch.somedom.com
50.100.200.in-addr.arpa    name server = thatch.somedom.com
thatch.somedom.com         internet address = 200.100.50.1
```

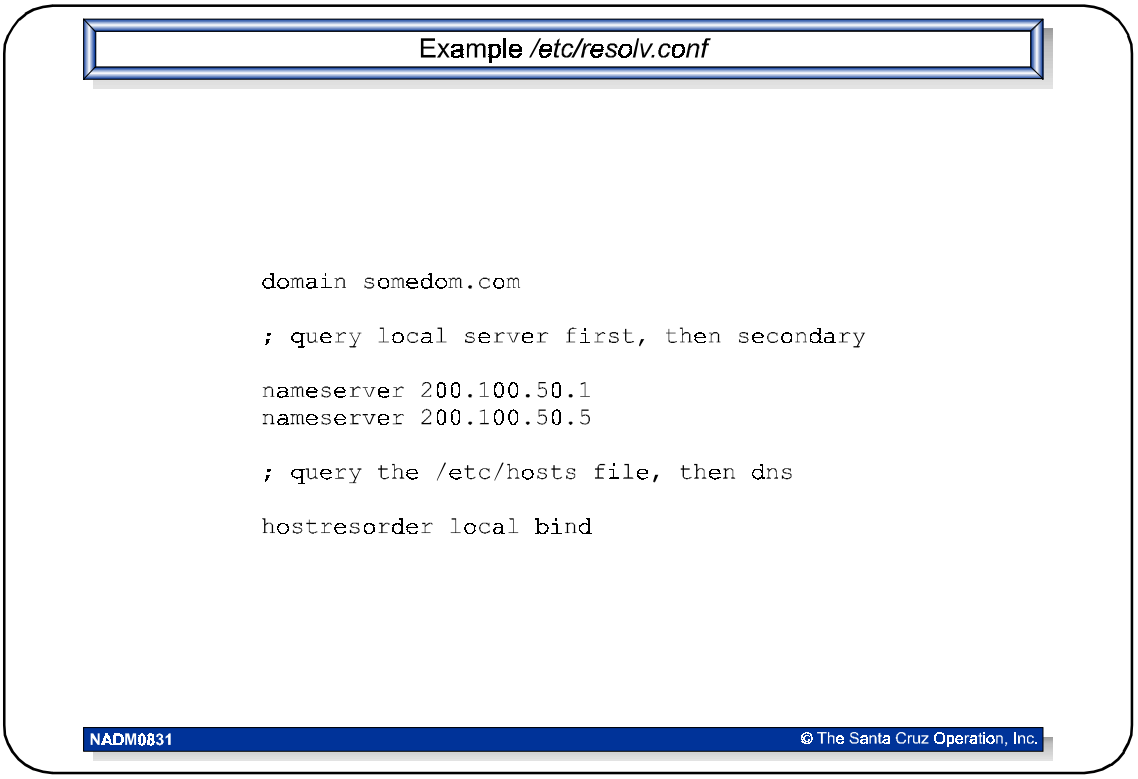
AV 8-29: The dig(ADMN) Command
Use this slide to illustrate what students should expect to see when querying with **dig**.

8.6.3 The host(ADMN) command

The **host** command is similar to **nslookup**(ADMN). It can be used to look up host names or IP addresses using DNS. When used with the **-a** option, it can be used to find all information about a host maintained by the domain server. The example in AV 8-30 shows how **host** is used to find all information about the server *www.sco.com*.

AV 8-30: The host(ADMN) Command
Use this slide to illustrate what students should expect to see when querying with **host**.

AV 8-31


Example /etc/resolv.conf
domain somedom.com
; query local server first, then secondary
nameserver 200.100.50.1
nameserver 200.100.50.5
; query the /etc/hosts file, then dns
hostresorder local bind
NADM0831
© The Santa Cruz Operation, Inc.

8.7 Configuring DNS Clients

After the name servers are configured, set up each host on the network as a DNS client. This includes systems configured as DNS name servers.

A DNS client is configured by creating the */etc/resolv.conf* file. This directs the resolver to name servers on the network to resolve queries. If a system is a name server, its *resolv.conf* file includes an entry for itself.

NOTE: A common mistake is to misspell the filename. Reinforce that it is *resolv.conf*, not *resolv**e**.conf*.

8.7.1 The */etc/resolv.conf* file

AV 8-31: Example */etc/resolv.conf*
This contains an example *resolv.conf* file that would be created on any of the clients in the example network described in this lesson.

The existence of the */etc/resolv.conf file* causes the resolver to look at this file to determine how to resolve queries. The basic format of this file is:

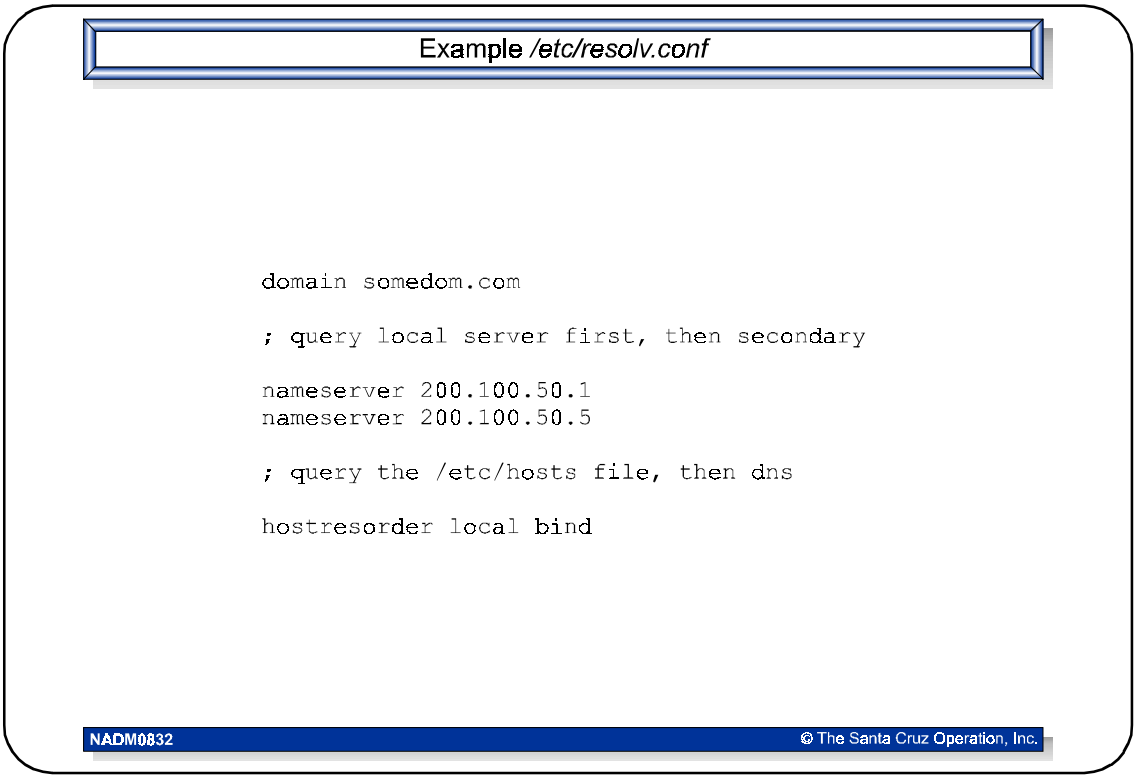
`domain`	Specifies the local domain name.
`name server`	Specifies the Internet address (in dot notation) of a name server that the resolver should query. Up to three name servers can be listed, one per keyword. If there are multiple servers, they are queried in the order listed.
`hostresorder`	Specifies database search order. The lookup routines can search any or all of the following databases: DNS, NIS, and the */etc/hosts* file. The `hostresorder` keyword value can include any or all of the tokens that represent these databases: `bind`, `nis`, or `local`. Note that searching the */etc/hosts* file prior to either of the other databases can lead to inconsistent results if the hosts file contains out-of-date information. However, avoiding network traffic during host name lookup can be useful when only demand-dial serial lines are available, because it can eliminate time and expense when resolving local names.

There are three other entries that are not in this example. They are used for these circumstances:

- `search` and `options` are used if you have set up a subdomain
- `sortlist` is used when you have subnets

See *resolv.conf*(SFF) for more information about these options.

AV 8-32


Example /etc/resolv.conf
domain somedom.com
; query local server first, then secondary
nameserver 200.100.50.1
nameserver 200.100.50.5
; query the /etc/hosts file, then dns
hostresorder local bind
NADM0832
© The Santa Cruz Operation, Inc.

AV 8-32: Example /etc/*resolv.conf*
The example *resolv.conf* file in AV 8-31 is repeated here.

AV 8-32 shows an example */etc/resolv.conf* file that you might create for any of the clients in the *somedom.com* network. The first `name server` entry points to the primary name server for *somedom.com* at IP address 200.100.50.1, which is *thatch.somedom.com* in our example.

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.

The example */etc/resolv.conf* file in AV 8-31 assumes that host 200.150.75.1 in *otherdom.com* is a secondary name server for *somedom.com*.

8.8 Lesson Summary

Name servers are 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 then on demand. Clients need only be pointed at a functioning name server to query data.

Name servers can forward queries for information they are not responsible for 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?

8. What commands can be used to query a name server?

8.9 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 is the primary name server
- Which hosts, if any, are secondary or caching-only servers

This lab consists of three parts:

- Configuring the name servers
- Testing the work completed in the first part
- Setting up and testing the clients

8.9.1 Configuring servers

Your instructor will provide you with the relevant data needed to create a primary, secondary, or caching-only server:

- If your host is a primary, secondary, or caching-only server, follow the procedure in the next section.
- If your host is a client, team up with one of the students to set up their server.

You have a lot of flexibility to set up servers and clients as you would like. However, to make the other labs work correctly, you must have at least one primary server, one secondary server, and one client.

Procedure: Configuring a primary name server

To configure a primary name server, create each of the DNS configuration files, substituting the names and IP addresses for your classroom provided by your instructor. Use the sample files in the */etc/named.d* directory and the example in Section 8.4.1 on page 8-57 as a guide.

Note that there is no sample *named.boot* file in Release 5.0.5. It has been replaced with the new format *named.conf* file. Samples of the files *named.local*, *named.rev*, *root.cache* and *named.soa* are located in */etc/named.d*.

Step 1. Create /*etc/named.boot* with entries for the primary name server.

Step 2. Create /*etc/named.d/named.hosts* with an A record for each host.

Step 3. Create /*etc/named.d/named.rev* with a PTR record for each host.

Step 4. Create /*etc/named.d/named.local* with a PTR record for *localhost.*

Step 5. If not already present, create /*etc/named/root.cache*. Copy this file from another server, or use **ftp** to transfer the latest version from *ftp.rs.internic.net*

Step 6. Create /*etc/named.d/named.soa* with an SOA record and NS records for the primary and secondary name servers.

Step 7. Create */etc/resolv.conf* with an entry for the primary name servers.

Step 8. Start **named** as **root**. Enter:

```
ndc start
```

If **named** is already running, signal the process to re-read the configuration files. Enter:

```
ndc restart
```

You can also use the command line:

```
kill -1 ‘cat /etc/named.pid‘
```

Test the server as described in Section 8.9.2 on page 8-77.

Procedure: Configuring a secondary name server

To configure a secondary name server, create each of the DNS configuration files described below, substituting the names and IP addresses for your classroom provided by your instructor. Use the sample files in the */etc/named.d* directory and the example in Section 8.4.2 on page 8-59 as a guide.

Step 1. Create */etc/named.boot* with entries for the secondary name server.

Step 2. Create */etc/named.d/named.local* with a PTR record for *localhost.*

Step 3. If not already present, create */etc/named/root.cache.* Copy this file from another server, or use **ftp** to transfer the latest version from *ftp.rs.internic.net*

Step 4. Create /*etc/named.d/named.soa* with an SOA record and NS records for the primary and secondary name servers.

Step 5. Create */etc/resolv.conf* with entries for the primary and any secondary name servers.

Step 6. Start **named** as **root**. Enter:

```
ndc start
```

If **named** is already running, signal the process to re-read the configuration files. Enter:

```
ndc restart
```

You can also use the command line:

```
kill -1 ‘cat /etc/named.pid‘
```

Test the server as described in Section 8.9.2 on page 8-77. Verify that the backup host and reverse lookup files are copied from the primary name server.

Procedure: Configuring a caching name server

To configure a caching name server, create the DNS configuration files, substituting the names and IP addresses for your classroom provided by your instructor. Use the sample files in the */etc/named.d* directory and the example in Section 8.4.3 on page 8-60 as a guide.

Step 1. Create */etc/named.boot*.

Step 2. Create */etc/named.d/named.local*.

Step 3. If not already present, create */etc/named.d/root.cache.* Copy this file from another server, or obtain the latest version from *ftp.rs.internic.net*.

Step 4. Create */etc/named.d/named.soa*.

Step 5. Create */etc/resolv.conf*.

Step 6. Start **named** as **root**. Enter:

```
ndc start
```

If **named** is already running, signal the process to re-read the configuration files. Enter:

```
ndc restart
```

You can also use the command line:

```
kill -1 ‘cat /etc/named.pid‘
```

Test the server as described in Section 8.9.2 on page 8-77.

8.9.2 Testing your server

The objective of this lab is for you to become familiar with utilities to test and query name servers. Verify that you can retrieve both the A and PTR records using either **dig**(ADMN) or **nslookup**(ADMN).

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.

Look up the names of at least two hosts on your network by entering in their IP addresses.

8.9.3 Configuring and testing a client

The objective of this lab is for you to configure a host as a DNS client which queries multiple name servers in case one is down, and to resort to querying the */etc/hosts* file if neither server is found.

Step 1. Create */etc/resolv.conf* with these types of entries, substituting the names and IP addresses for your classroom name servers:

```
name server 200.100.50.1
name server 200.100.50.5
name server 200.100.50.6
hostresorder local bind
```

If your host is also a name server, list your name server first.

Step 2. Use **nslookup** to query for the IP address of one or more of the machines on your LAN.

Step 3. Stop the name server on the host that you listed first in the */etc/resolv.conf* file.

Step 4. 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 5. Stop the name server on the host that you listed as your second choice in your */etc/resolv.conf* file.

Step 6. Try using a client such as **telnet**(TC) 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.

8.10 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

dig(ADMN)	Sends domain name query packets to name servers
host(ADMN)	Looks up host names using DNS
named(ADMN)	Implements BIND
nslookup(ADMN)	Queries name servers

Relevant files

/etc/named.boot	Default file read by **named.** Refer to the *named.boot*(SFF) manual page.
/etc/named.conf	New format **named** configuration file for BIND 8.1.1**.** Refer to the *named.conf*(SFF) manual page.
/etc/named.d	Default directory for locating **named** database files in SCO OpenServer Release 5.
/etc/resolv.conf	Contains the *domain* and *name server* entries to point the client at name server. Refer to the *resolv.conf*(SFF) manual page.
/usr/lib/named	Default directory for locating **named** database files in SCO operating systems prior to SCO OpenServer Release 5.
/etc/inet/named.d	Default directory for locating **named** database files in UnixWare 7.
named.hosts	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 *named.hosts*(SFF) manual page.

root.cache	Default name for the file containing the A and NS records for the root name servers. Refer to the *root.cache*(SFF) manual page.
named.local	Default name of the file containing the PTR record for *localhost*. Refer to the *named.local*(SFF) manual page.
named.rev	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 *named.rev*(SFF) manual page.
named.soa	Default name for the file containing the SOA record and NS records for a zone. Refer to the *named.soa*(SFF) manual page.

8.11 Answers to Review Questions

This section provides suggested 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 found 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 *named.hosts*.

5. CNAME records allow you to create nicknames or aliases for hosts. They are found in the host database files such as *named.hosts*.

6. PTR records map IP addresses to names. They are typically found in the reverse lookup, *in-addr.arpa* database files such as *named.rev.*

7. The */etc/resolv.conf* file must be created on all client machines to point the resolver at a name server.

8. The **nslookup**(ADMN), **host**(ADMN), and **dig**(ADMN) commands can be used to query a name server.

9

Configuring Internet Services

Instructor Guide

Lesson 9

Configuring Internet Services

Suggested lesson duration

This lesson should take approximately 3 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

Configure your classroom name server to act as a primary server for the (fictitious) *vd1.org* domain. Create A records for each machine in the classroom with an IP address the students will use to create virtual domains and configure virtual-domain services on their machines. When the lab begins, give each student the IP address and name of their virtual machine.

Determine which primary and secondary NTP servers to poll from.
http://www.eecis.udel.edu/~mills/ntp/servers.html.

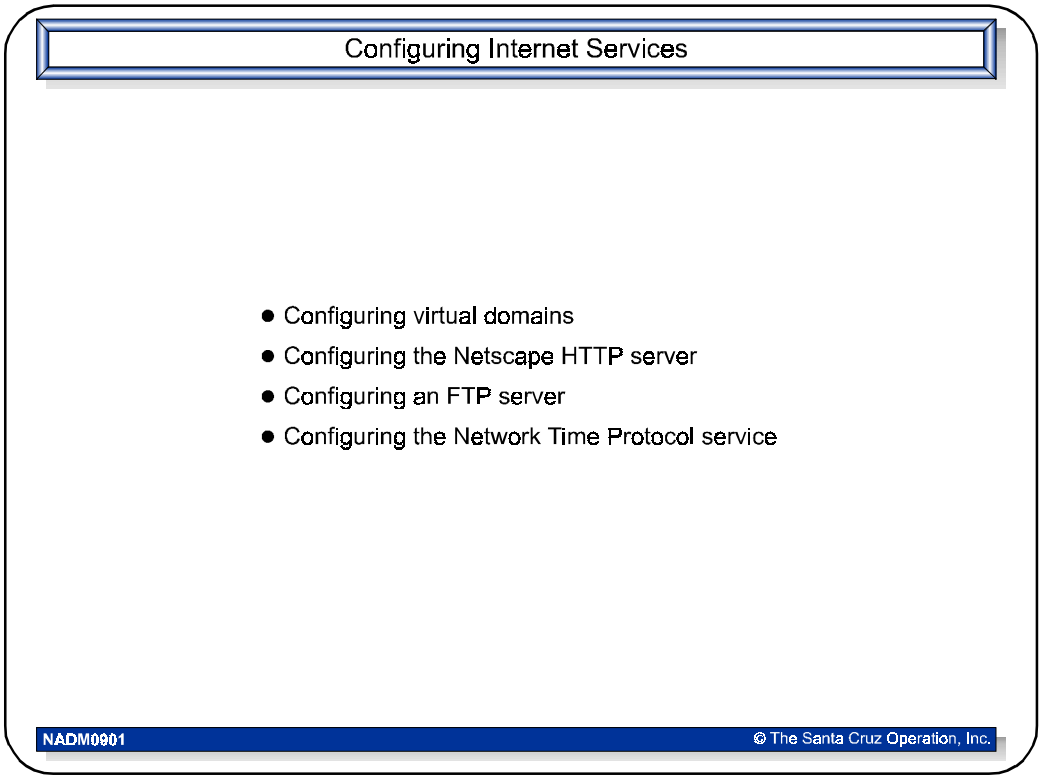
Instructor preparation

If the machine's name or IP address was changed during previous labs, change the Netscape™ authorizations to include this machine's current name or IP address.

The instructor should be fully versed in virtual domains discussed in **scohelp ⇨ SCO Internet Family Documentation ⇨ SCO Internet FastStart Administrator's Guide ⇨ Chapter 3: Configuring Internet services ⇨ Managing virtual domains.**

The instructor should also be familiar with Netscape Server configuration which is discussed in **scohelp ⇨ SCO Internet Family Documentation ⇨ Netscape FastTrack Server Administrator's Guide.**

AV 9-1


Configuring Internet Services
Configuring virtual domains
Configuring the Netscape HTTP server
Configuring an FTP server
Configuring the Network Time Protocol service
NADM0901
© The Santa Cruz Operation, Inc.

Introduction

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 found using machines running TCP/IP on local and wide area networks.

SCO OpenServer Release 5 provides several Internet-standard services that make a network more usable. Understanding how to configure and use these services will enhance the value of your company's network and make your job as a network administrator easier.

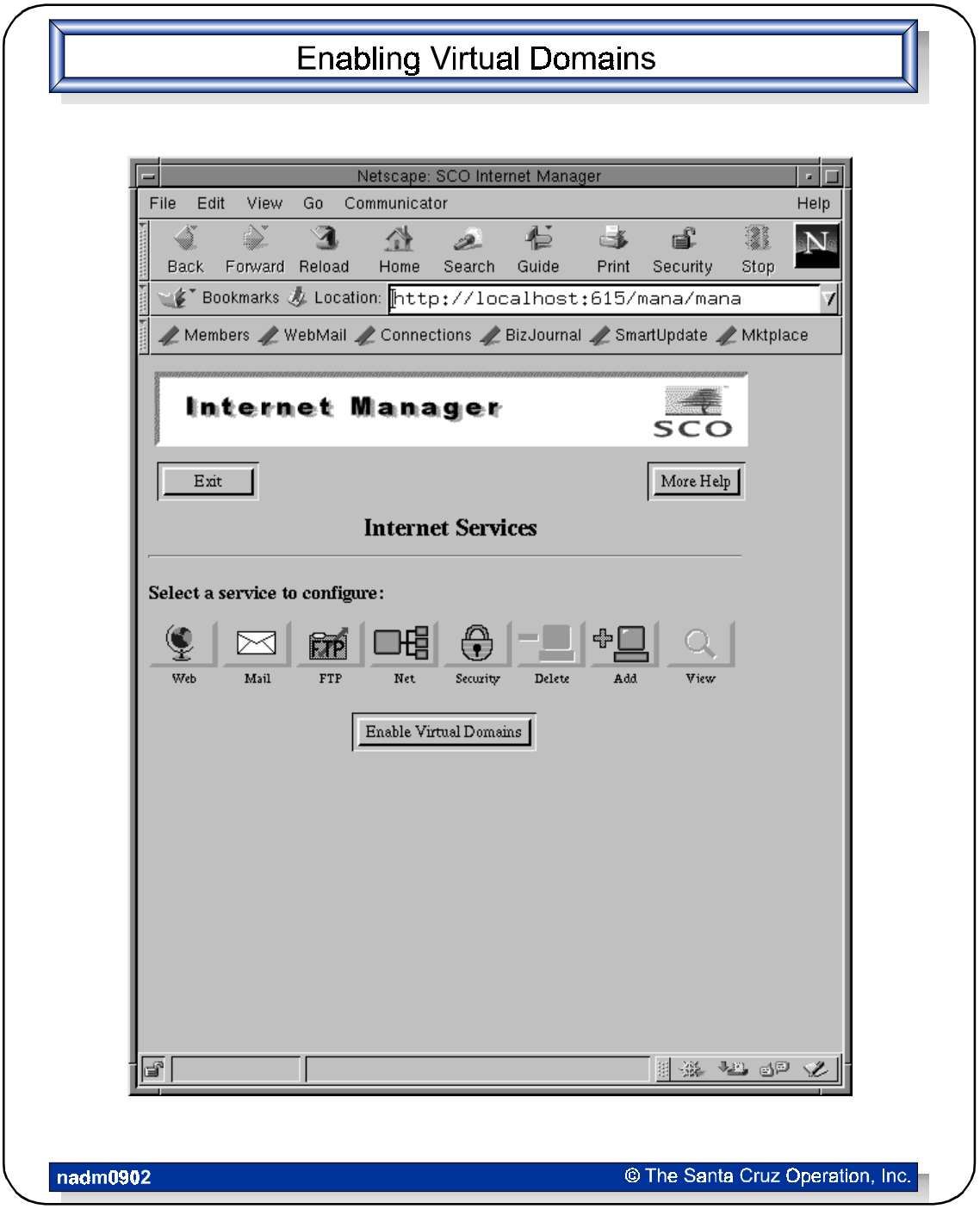
Objectives

This lesson describes the standard Internet services available on SCO OpenServer Release 5. By the end of this lesson, you will be able to:

- Create a virtual domain
- Describe the relationship between virtual domains and other services
- Start the Netscape HTTP server and enable remote administration
- Configure FTP for normal and anonymous access
- Configure the Network Time Protocol (NTP) server and clients

AV 9-1: Configuring Internet Services
This first slide describes the objectives covered in this lesson.

AV 9-2


Enabling Virtual Domains
Netscape: SCO Internet Manager
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Exit
More Help
Internet Services
Select a service to configure:
Web Mail FTP Net Security Delete Add View
Enable Virtual Domains
nadm0902
© The Santa Cruz Operation, Inc.

9.1 Virtual Domains

You can configure your machine so that it provides e-mail, FTP, and web services for several different domain names. For example, you might want to:

- Sell an *Internet hosting* service to other organizations
- Provide the service as a favor to a non-profit community organization

These services can be provided using a feature known as *virtual domains*. In reality, there is no such thing as a virtual domain, only virtual hosts. However, because all of the literature, online documentation, and tools use the term virtual domain, this course will too. It might help to think of a virtual domain as a virtual host residing on your computer.

9.1.1 Enabling virtual domains

AV 9-2: Enabling Virtual Domains
This slide shows the **Internet Manager Internet Services** screen before enabling virtual domains.

Prior to enabling virtual domains, your system's network services process all requests equally, independent of which network interface receives the request.

For example, the World Wide Web server does not care whether a request is received through a PPP connection or the system's network card. After virtual domains are enabled, the system's network services are configured independently, and can behave differently depending on the network interface.

To enable a virtual domain:

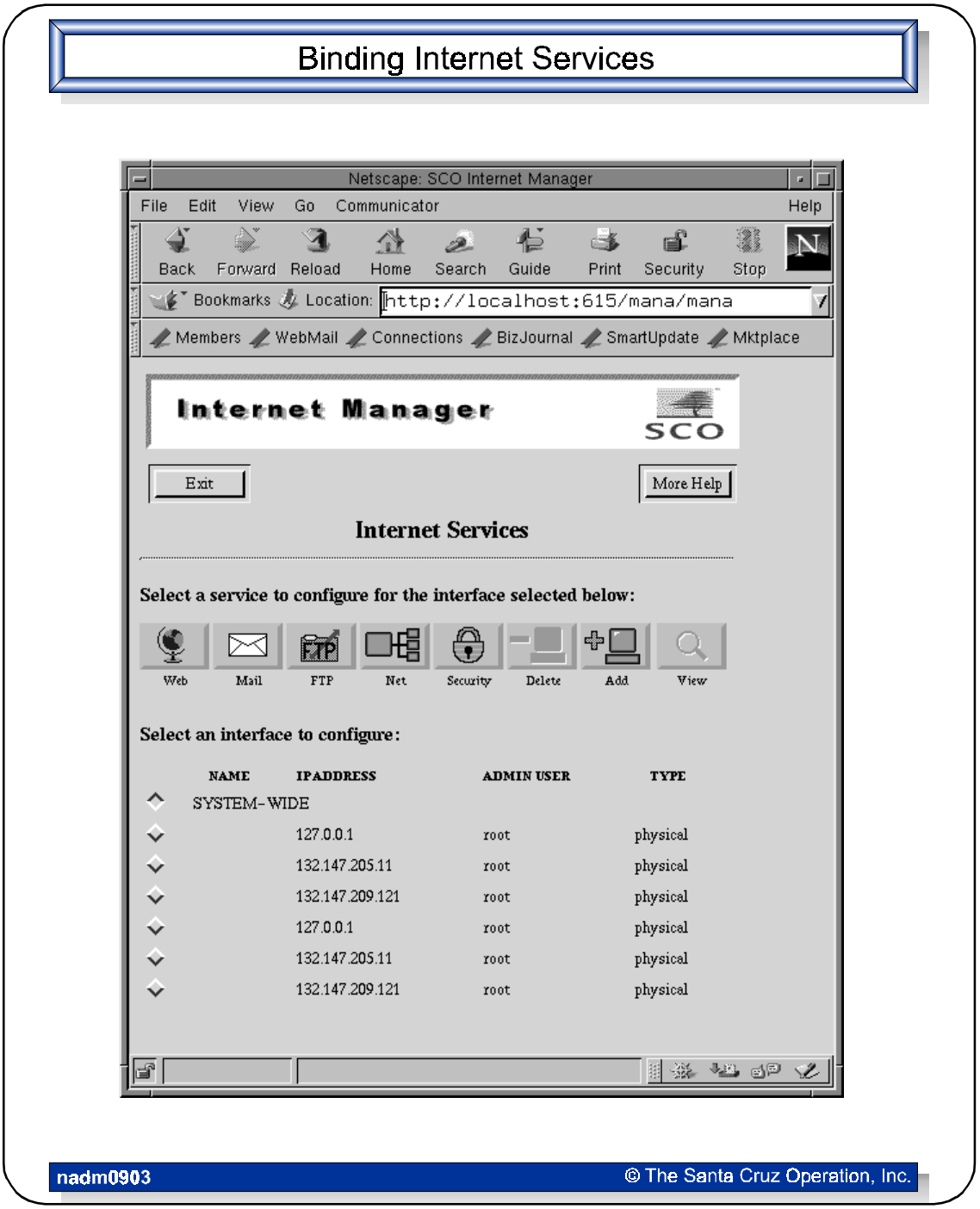
Step 1. Start the **Internet Manager**.

Step 2. Log in using the account name **admin** with the **root** password.

Step 3. On the **Internet Services** page, select **Enable Virtual Domains** and click on **Yes** ungroup the networking interfaces.

Ungrouping network interfaces allows you to configure virtual domains. NOTE: After the interfaces are ungrouped, they cannot be reconfigured as a group again.

AV 9-3


Binding Internet Services
Netscape: SCO Internet Manager
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Exit
More Help
Internet Services
Select a service to configure for the interface selected below:
Web Mail FTP Net Security Delete Add View
Select an interface to configure:
NAME IPADDRESS ADMIN USER TYPE
SYSTEM-WIDE
127.0.0.1 root physical
132.147.205.11 root physical
132.147.209.121 root physical
127.0.0.1 root physical
132.147.205.11 root physical
132.147.209.121 root physical
nadm0903
© The Santa Cruz Operation, Inc.

Step 4. Choose a network interface to bind the Internet services. Use the IP address of the system (that is, the IP number by which the system is already known).

Choosing a network interface (such as *net0*) rather than another interface, such as *localhost*, can simplify routing. If you use IP addresses for virtual domains that are on the same subnet as the network interface chosen here, you will not have to make changes on your router.

After this form is submitted, the **Internet Manager** displays a list of networking interfaces.

AV 9-3: Binding Internet Services
This slide shows the **Internet Manager** displaying each network interface after enabling virtual domains. One of these, *132.147.209.121* in this example, is selected to bind Internet services.

Exercise: Enabling virtual domains

To enable virtual domains:

Step 1. Start the **Internet Manager**.

Step 2. Log in using the account name **admin** with the **root** password.

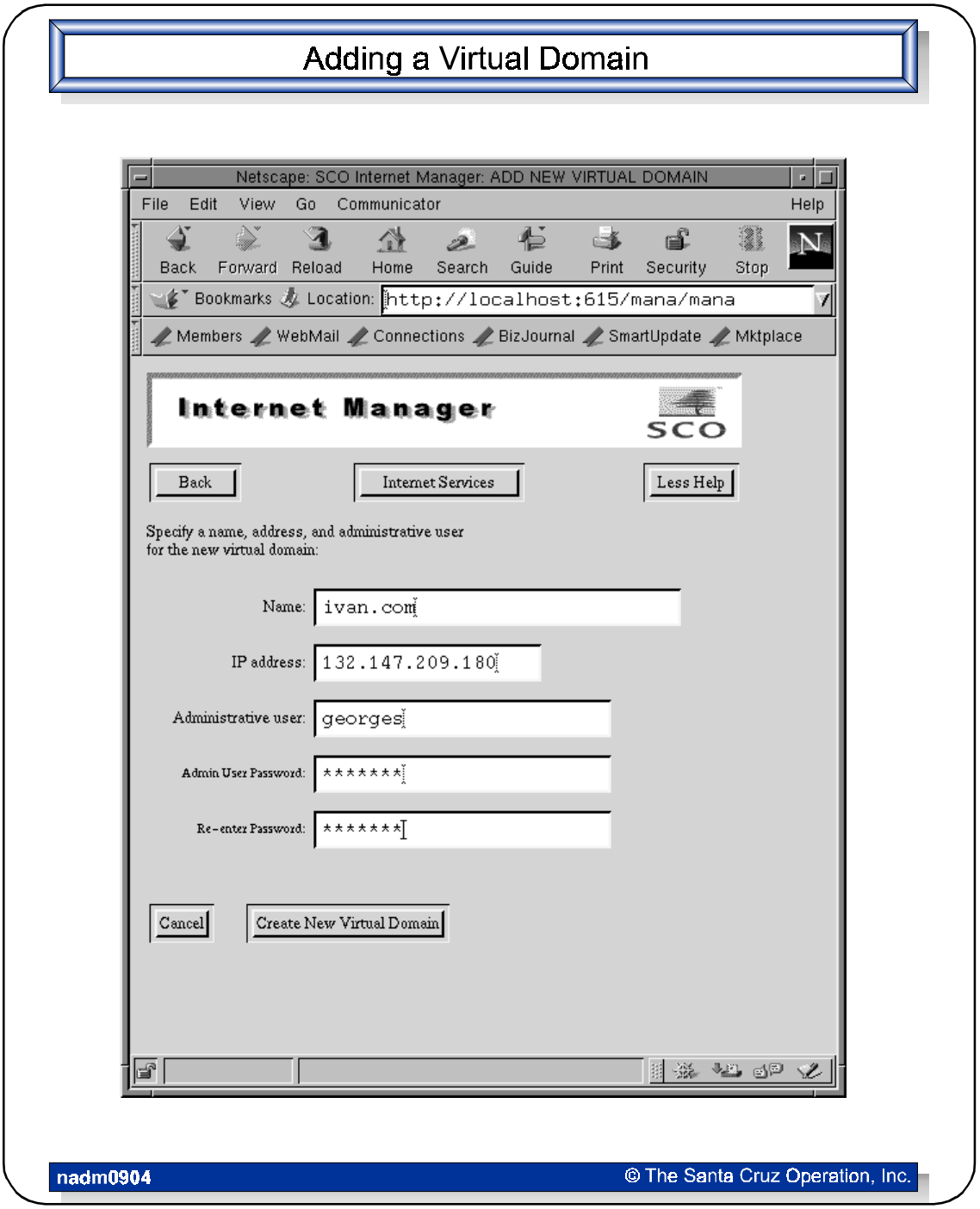
Step 3. On the **Internet Services** page, select **Enable Virtual Domains**.

Step 4. When prompted, confirm with **Yes**.

Step 5. Choose a network interface to bind the Internet services to. Use the IP address for your LAN interface.

Step 6. Examine the directories in */usr/internet/ip* that are created when the interfaces are ungrouped.

AV 9-4


Adding a Virtual Domain
Netscape: SCO Internet Manager: ADD NEW VIRTUAL DOMAIN
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Back
Internet Services
Less Help
Specify a name, address, and administrative user for the new virtual domain:
Name: ivan.com
IP address: 132.147.209.180
Administrative user: georges
Admin User Password: *******
Re-enter Password: *******
Cancel
Create New Virtual Domain
nadm0904
© The Santa Cruz Operation, Inc.

9.1.2 Creating a virtual domain

After virtual domains are enabled (Section 9.1.1 on page 9-7), you can create a virtual domain:

Step 1. Start the **Internet Manager**.

Step 2. Log in using the account name **admin** with the **root** password.

Step 3. On the **Internet Services** page, select the **Add** icon and fill in the form shown in AV 9-4:

Name — Enter a unique, fully-qualified, domain name for the new virtual domain. If this virtual domain is accessible on the Internet, obtain a registered domain name from the InterNIC.

IP address — Enter a unique IP address for this interface. If possible, use an IP address in the same subnet as the primary network interface to simplify routing.

If this virtual domain is accessible on the Internet, obtain a registered IP address.

Administrative user — Choose a name for the administrator for this domain. This is the name used to access the **Internet Manager** configuration tools.

Admin User Password — Enter a password for the administrative user account.

AV 9-4: Adding a Virtual Domain
This slide shows the **Internet Manager Internet Services** used to add a virtual domain.

Step 4. Click on **Create New Virtual Domain**.

The **Internet Manager** adds the new user account and creates a directory hierarchy for the virtual domain in */usr/internet/ip*. This directory contains files and directories for each of the Internet services that can be configured for a virtual domain.

Each virtual host is displayed twice. This is an anomaly of the **Internet Manager** interface.

Exercise: Adding virtual domains

This exercise assumes that virtual domains are enabled.

Step 1. Select **Add** from the **Internet Services** page of the **Internet Manager**.

Step 2. Fill in the fully-qualified domain name and administrative user information.

Step 3. Click on **Create New Virtual Domains**.

Step 4. Examine the newly-created directories and files in */usr/internet/ip/a.b.c.d* (where *a.b.c.d* is the IP address of the virtual domain).

Step 5. Examine the new user account in */etc/passwd*.

Step 6. Connect to the web server for this new virtual domain using *http://a.b.c.b:80/* as the URL.

9.1.3 Setting up 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 multiple domains. General DNS configuration is discussed in Lesson 8: “Domain Name System (DNS) Configuration.”

E-mail

Configuring e-mail services for virtual domains is supported using the **sendmail**(ADMN) Mail Transport Agent. In addition to configuring e-mail for virtual domains, you will need to set up MX records for your virtual domain in DNS.

Configuring e-mail for virtual domains is beyond the scope of this course. For more information, refer to **scohelp ⇨ SCO Internet Family Documentation ⇨ SCO Internet FastStart Administrator's Guide ⇨ Chapter 3: Configuring Internet services ⇨ Managing virtual domains ⇨ Viewing properties of virtual domains ⇨ Configuring e-mail.**

Web

The Netscape FastTrack Server can be configured for virtual domains. An overview of Netscape server administration is provided in the next section of this lesson.

Detailed configuration of web services is beyond the scope of this course. For further information, see **scohelp ⇨ SCO Internet Family Documentation ⇨ Netscape FastTrack Server Administrator's Guide ⇨ Chapter 3: Configuring Internet services ⇨ Setting up software virtual servers**.

FTP

Anonymous FTP, described later in this lesson, can be configured for virtual domains. For further information, see **scohelp ⇨ SCO SCO Internet Family Documentation ⇨ SCO Internet FastStart Administator's Guide ⇨ Chapter 3: Configuring Internet services ⇨ FTP**.

9.2 Web Services

The Internet has experienced explosive growth for two main reasons:

- A gigantic amount of information is available.

To demonstrate the amount of information available, poll the students for a list of their favorite web sites and share a few of your own.

- 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 SCO OpenServer Release 5 system profiles. This is the server described in this course.

- Apache—available on Skunkware. More information is available at *http://www.apache.org*.

- NCSA HTTPd—included with SCO OpenServer to serve **scohelp** and manual pages. It runs by default on port 457.

9.2.1 Configuring the Netscape FastTrack Server

The FastTrack web server is started during system boot and runs on port 80. This server is used by default when web browsers contact your host.

The Netscape **Administration Server** is another web server used to configure the FastTrack server. It is started from **scoadmin**(ADM) and runs on port 620. The **Administration Server** allows you to add, remove, and configure existing web servers.

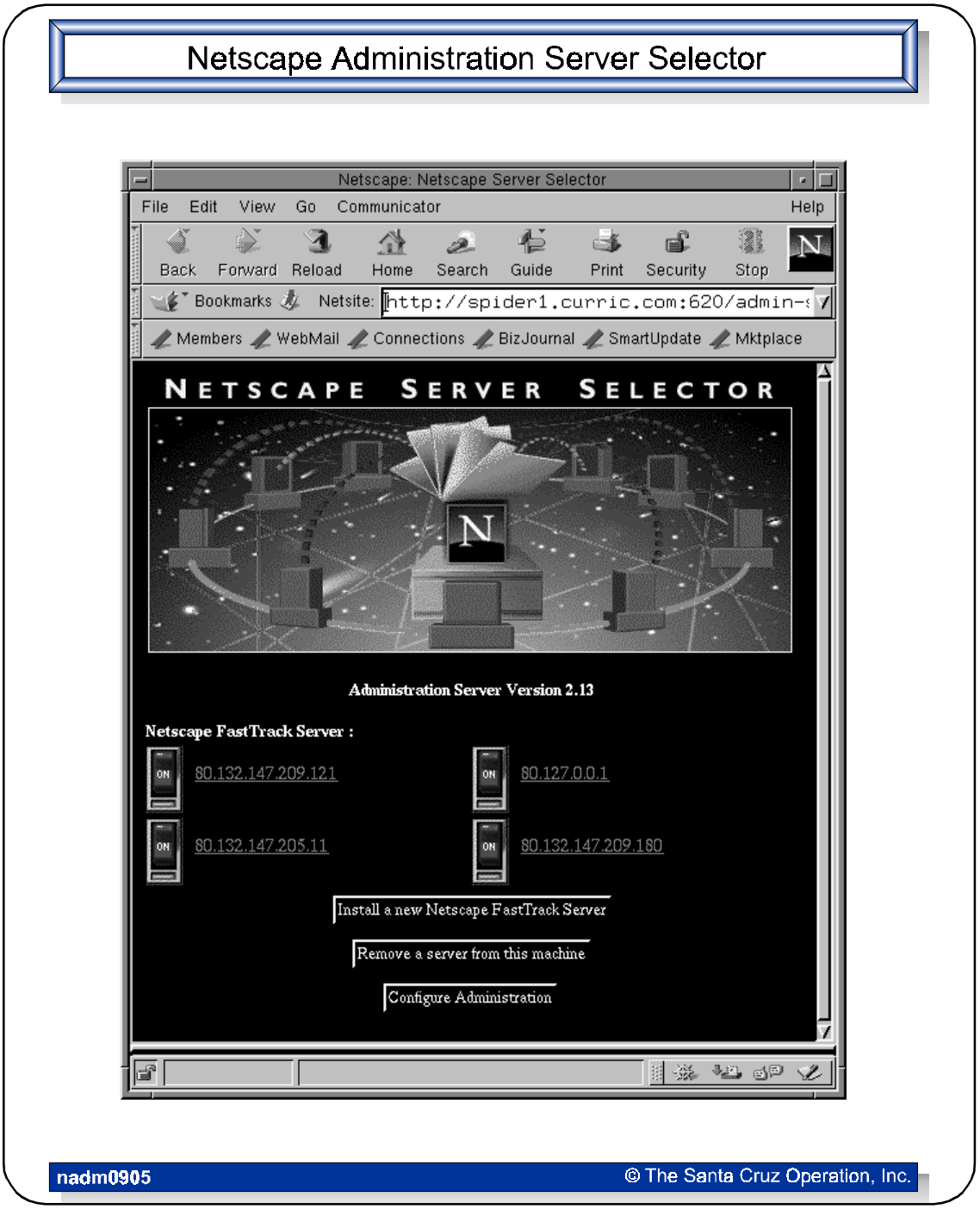
From the Netscape **Administration Server**, you can configure these features of your FastTrack server:

System Settings	Sets information about the FastTrack Server including the directory location of server binaries, the hostname 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 ⇨ SCO Internet Family Documentation ⇨ Netscape FastTrack Server Administrator's Guide.**

AV 9-5


Netscape Administration Server Selector
Netscape: Netscape Server Selector
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Netsite: http://spider1.curric.com:620/admin-s
Members WebMail Connections BizJournal SmartUpdate Mktplace
NETSCAPE SERVER SELECTOR
Administration Server Version 2.13
Netscape FastTrack Server :
ON 80.132.147.209.121
ON 80.127.0.0.1
ON 80.132.147.205.11
ON 80.132.147.209.180
Install a new Netscape FastTrack Server
Remove a server from this machine
Configure Administration
nadm0905
© The Santa Cruz Operation, Inc.

9.2.2 Running the Netscape Administration Server

AV 9-5: Netscape Administration Server Selector
This slide shows the **Administration Server Selector** page.

To administer the Netscape FastTrack server:

Step 1. Start the **Administration Server** from **scoadmin ⇨ Netscape ⇨ Netscape Server Admin**.

This also starts the Netscape Navigator browser with URL *http://hostname:620* where *hostname* is the name of your system. As shown in AV 9-5, this is called the **Server Selector**.

Step 2. Choose a server to configure. The Netscape FastTrack Server (port 80) is displayed for each networking interface.

Step 3. Enter the user name **admin** and the password for the **root** account.

This password is the password assigned to **root** during the system installation. This password does not change when you change **root**'s password. It can be changed using the **Administration Server** after entry is gained.

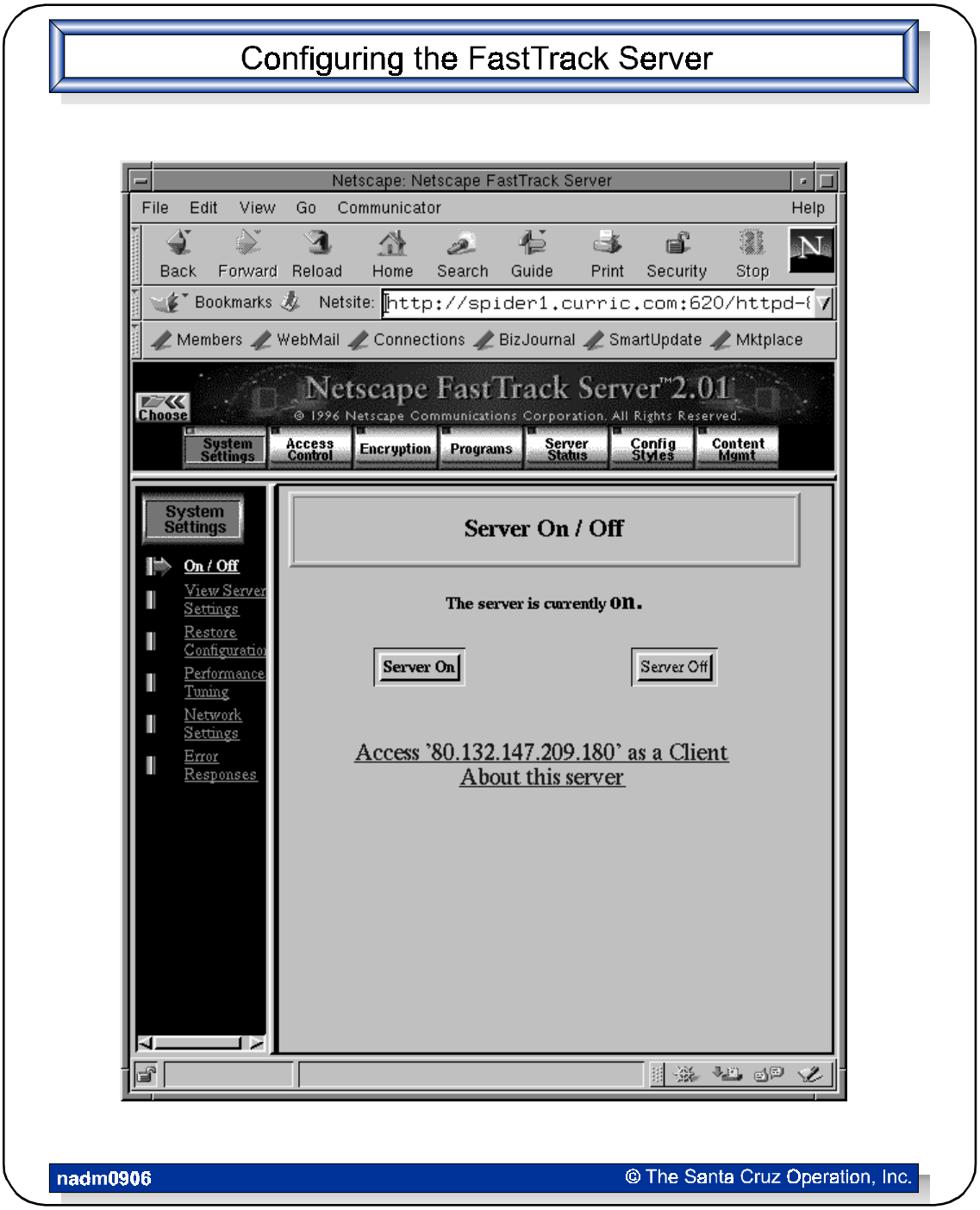
AV 9-5 shows the home page of the **Administration Server** for the selected FastTrack server.

Step 4. You can now make changes to the selected server configuration. Be sure to follow the prompts to stop and restart the server after making changes.

Step 5. Reload URL *http://hostname:620* to return to the **Server Selector**.

NOTE: If you cannot remember the **admin** password, delete the password by editing */usr/internet/ns_httpd/admserv/admpw* and deleting everything after the colon (:) character.

AV 9-6


Configuring the FastTrack Server
Netscape: Netscape FastTrack Server
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Netsite: http://spider1.curric.com:620/httpd-
Members WebMail Connections BizJournal SmartUpdate Mktplace
Netscape FastTrack Server™2.01
© 1996 Netscape Communications Corporation. All Rights Reserved.
Choose
System Settings
Access Control
Encryption
Programs
Server Status
Config Styles
Content Mgmt
System Settings
On / Off
View Server Settings
Restore Configuratio
Performance Tuning
Network Settings
Error Responses
Server On / Off
The server is currently on.
Server On
Server Off
Access '80.132.147.209.180' as a Client
About this server
nadm0906
© The Santa Cruz Operation, Inc.

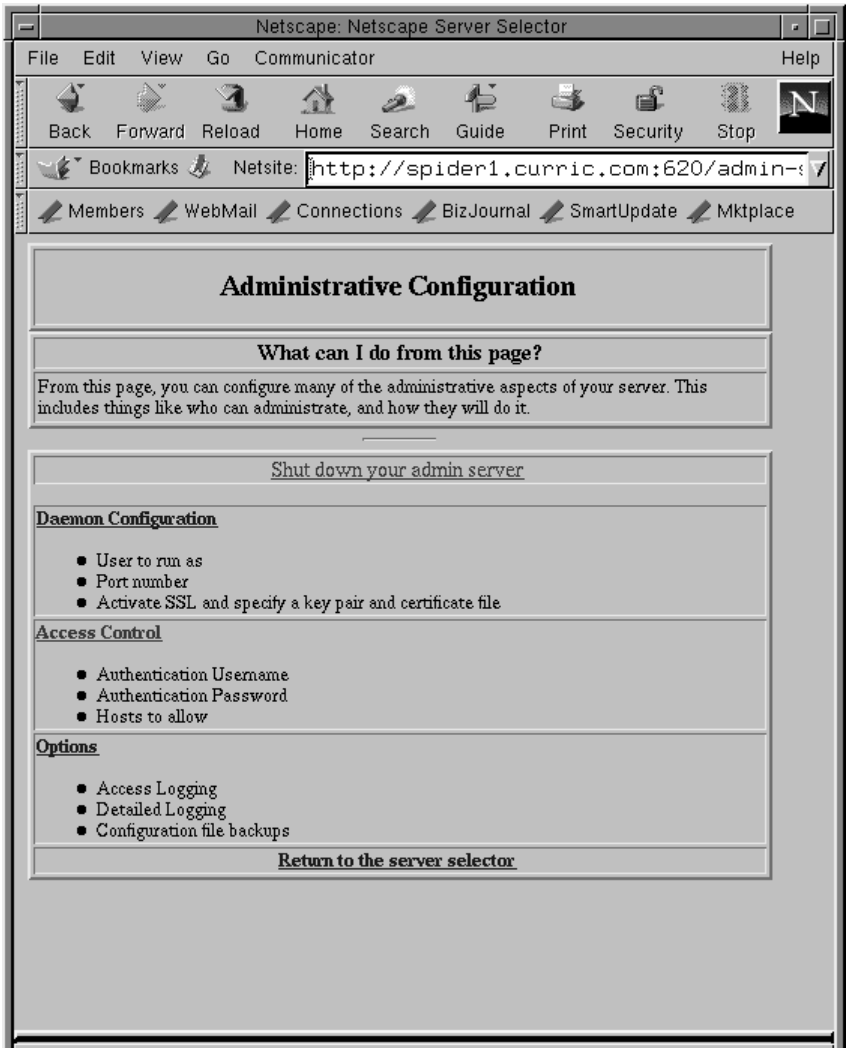
AV 9-6: Configuring the FastTrack Server
This slide shows the **Administration Server** configuration page for a specific FastTrack server. This course does not cover detailed server configuration. This is just an overview to make students aware of the tools used to configure the Netscape FastTrack server.

Exercise: Running the Netscape Administration Server

Step 1.	Start the Netscape **Administration Server**.
Step 2.	Select a server to configure.
Step 3.	Enter the user name **admin** and the password for the **root** account.
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:*a.b.c.d*' as a Client**. This opens a browser window and displays the root home page for the server with the displayed IP address.
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.	Access the same URL (using the **Reload** button). You can no longer access the URL.
Step 9.	Click **Server On** to restart the server in the **Administration Server** window.
Step 10.	Reload the URL again to verify that the server is running.
Step 11.	Return to the **Server Selector** using the URL *http://hostname:620.*

AV 9-7

Netscape Server Selector-Configuring Administration



nadm0907 © The Santa Cruz Operation, Inc.

9.2.3 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 **telnet** to the machine running the **Administration Server**.

Make it clear that this guide covers the use of the graphical and command-line interfaces that can be used to configure a server to be administered remotely. It does not discuss remote command-line administration of a server.

9.2.3.1 Configuring remote access using the Administration Server

AV 9-7: Netscape Server Selector—Configuring Administration
This slide shows the **Configure Administration** page of the Netscape **Administration Server**.

To configure remote administration:

Step 1. Start the **Administration Server** from **scoadmin ⇨ Netscape ⇨ Netscape Server Admin**.

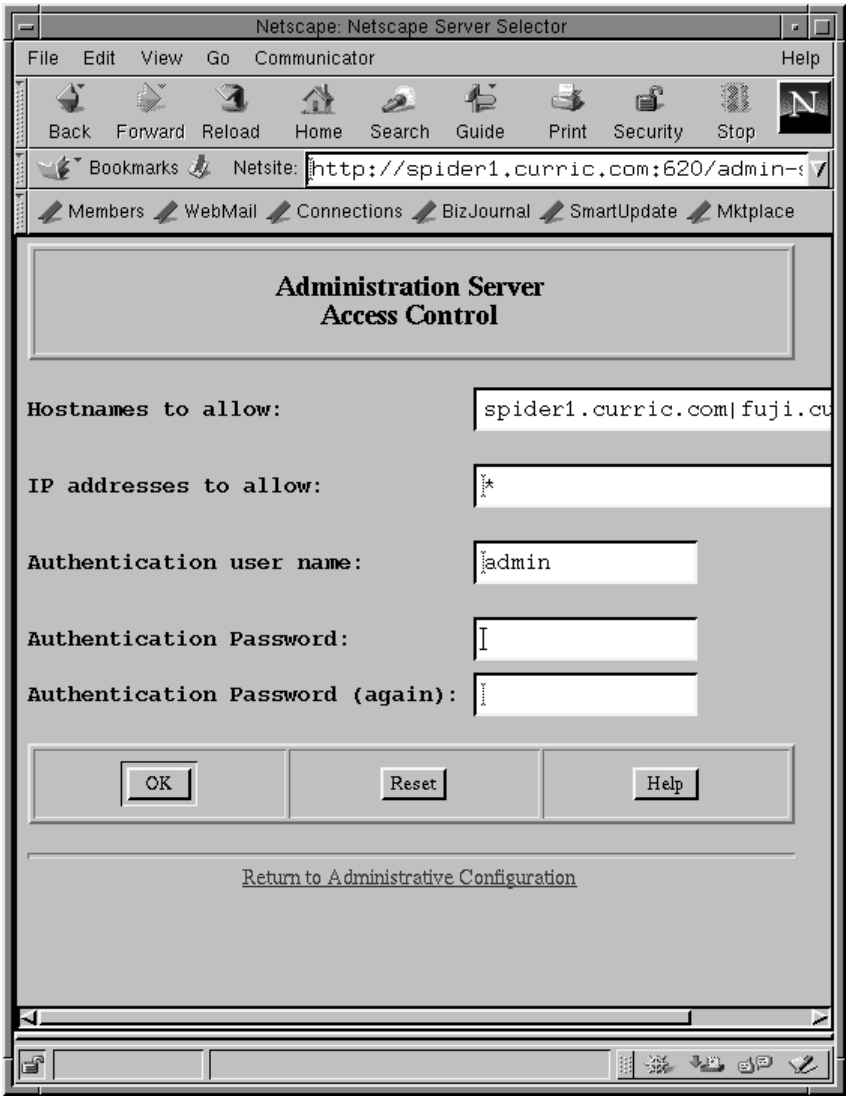
This also starts the Netscape Navigator browser with URL *http://hostname:620* where *hostname* is the name of your system. As shown in AV 9-5, this is called the **Server Selector**.

Step 2. From the **Server Selector**, select the **Configure Administration** button.

Step 3. Select “Access Control” link from the Administration Configuration page.

AV 9-8

Configuring Administration-Access Control



nadm0908 © The Santa Cruz Operation, Inc.

AV 9-8: Configuring Administration—Access Control
Use this slide to show students how to modify access control for Netscape Server Administration from remote sites.

Step 4. 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 5. Click on **OK**. The **Success!** page lets you know that your changes have been made.

Step 6. Click on the "shut it down" link, then click on the button **Shut down the administrative server**.

Step 7. Restart the **Administration Server** as **root**: Enter:

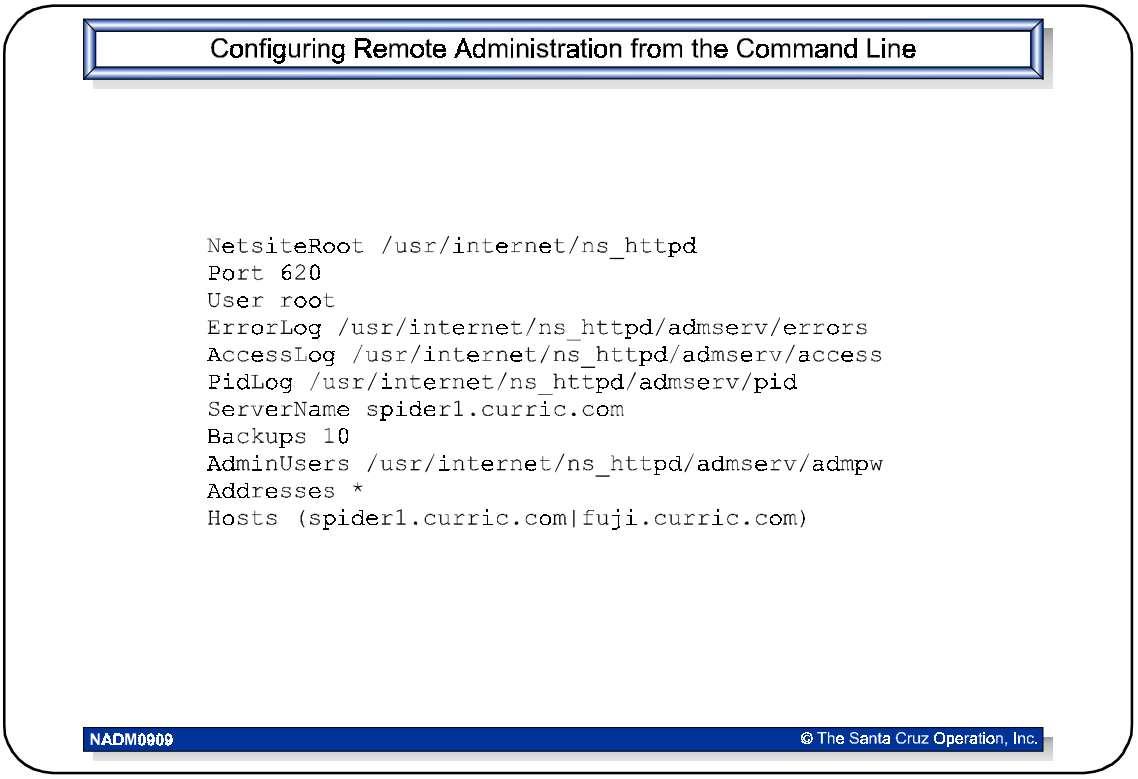
```
/usr/internet/ns_httpd/start-admin
```

Step 8. Verify the changes. Access the **Administration Server** from both the local and the remote hosts.

WARNING: If you misconfigure remote access, you can prevent any access to the **Administration Server**, even from the local host.

It is advised that students try both hosts to make sure that they have not denied access to their own host.

AV 9-9


Configuring Remote Administration from the Command Line
NetsiteRoot /usr/internet/ns_httpd
Port 620
User root
ErrorLog /usr/internet/ns_httpd/admserv/errors
AccessLog /usr/internet/ns_httpd/admserv/access
PidLog /usr/internet/ns_httpd/admserv/pid
ServerName spider1.curric.com
Backups 10
AdminUsers /usr/internet/ns_httpd/admserv/admpw
Addresses *
Hosts (spider1.curric.com|fuji.curric.com)
NADM0909
© The Santa Cruz Operation, Inc.

9.2.3.2 Configuring remote access from the command line

AV 9-9: Configuring Remote Administration from the Command Line
This slide shows an example FastTrack Server configuration file, */usr/internet/ns_httpd/admserv/ns-admin.conf*.

To configure remote administration using the command line:

Step 1. Log in as **root**.

Step 2. Edit the */usr/internet/ns_httpd/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/internet/ns_httpd/stop-admin
```

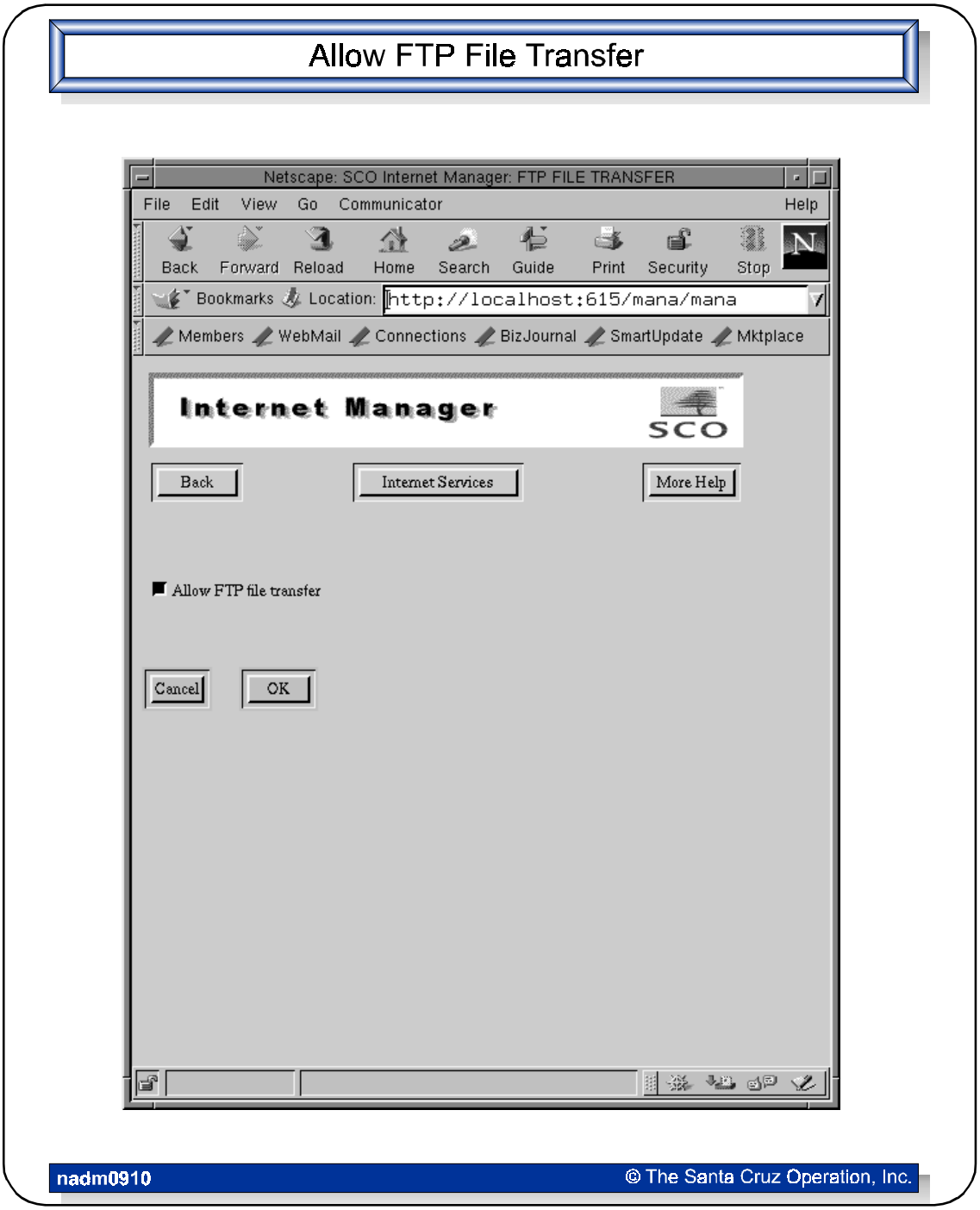
```
/usr/internet/ns_httpd/start-admin
```

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*.

Tools available to analyze the usage of your web server, such as **wusage**, **wwwstat** and **SurfReport**, can be obtained from:

- *http://www.boutell.com/wusage/download.html*
- *http://www.ics.uci.edu/websoft/wwwstat*
- *http://www.netrics.com*

AV 9-10


Allow FTP File Transfer
Netscape: SCO Internet Manager: FTP FILE TRANSFER
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Members WebMail Connections BizJournal SmartUpdate Mktplace
Internet Manager
SCO
Back
Internet Services
More Help
Allow FTP file transfer
Cancel
OK
nadm0910
© The Santa Cruz Operation, Inc.

9.3 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 SCO OpenServer Release 5 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.

9.3.1 Enabling FTP

AV 9-10: Allow FTP File Transfer
This slide shows the **FTP File Transfer** configuration screen in the **Internet Manager**.

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**(ADMN), 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. Start the **Internet Manager** as **root**.

Step 2. Select "SYSTEM-WIDE" as the networking interface to configure if you have enabled virtual domains.

Step 3. Click on the **FTP** icon. This displays the window shown in AV 9-10.

Step 4. Select "Allow FTP file transfer" and click on **OK**.

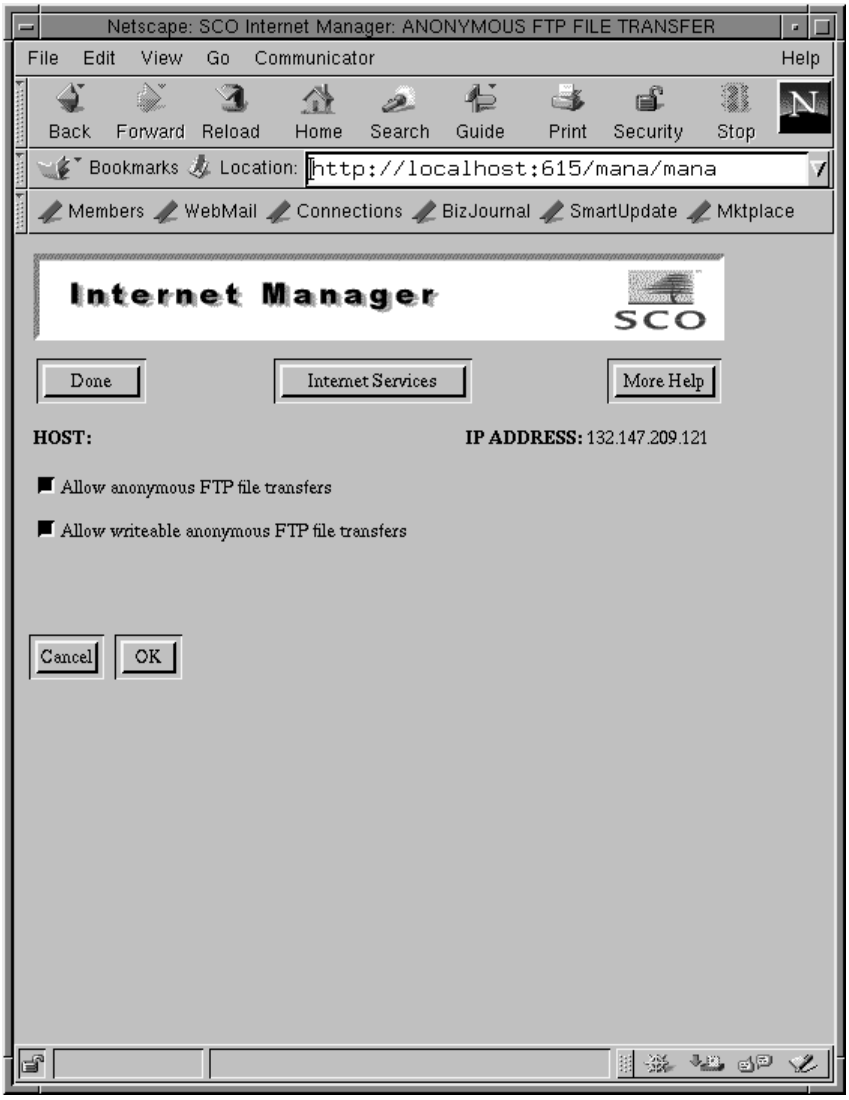
Exercise: Enabling FTP

Use the steps above to enable FTP access to your system.

Use **ftp**(TC) to connect to your server using an existing user account (such as **root**).

AV 9-11

Configuring Anonymous FTP



nadm0911 © The Santa Cruz Operation, Inc.

9.3.2 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 **Internet Manager**.

Step 2. If you have not already done so, enable FTP as described in Section 9.3.1 on page 9-27.

Step 3. Select the networking interface to configure.

Step 4. Click on the **FTP** icon. This displays the window shown in AV 9-11.

Step 5. Select "Allow anonymous FTP file transfers" to allow others to download files from the directory hierarchy for the selected interface.

Step 6. Select "Allow writable anonymous FTP file transfers" to allow others to upload files to the incoming FTP directory for the selected interface.

Step 7. Click on **OK**.

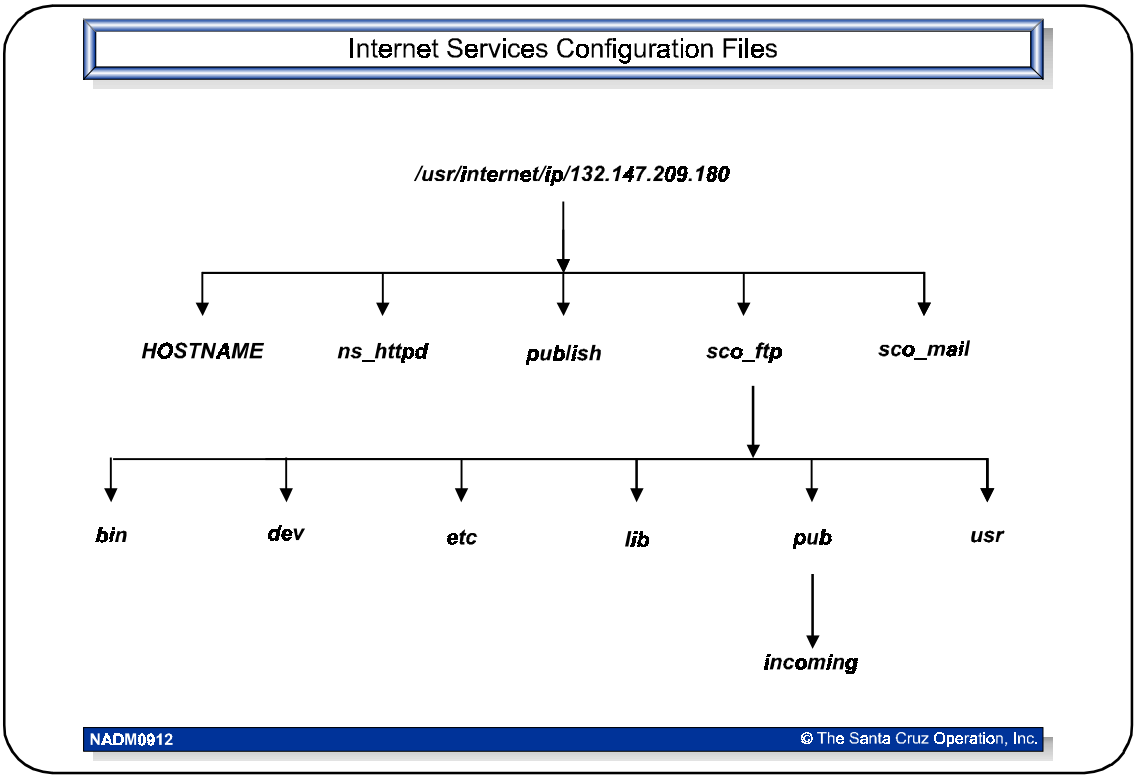
Step 8. When finished, the **Internet Manager** creates a file structure for FTP in the server's configuration directory, */usr/internet/ip/a.b.c.d* (where *a.b.c.d* is the IP address of the selected interface). See AV 9-14 on page 9-42.

If virtual domains are not configured, the directory is */usr/internet/ip/0.0.0.0*.

AV 9-11: Configuring Anonymous FTP
This slide shows the **Anonymous FTP File Transfer** configuration screen in the **Internet Manager**.

AV 9-12: Internet Services Configuration Files
The slide, on the next page, shows the files and directories created when anonymous FTP is configured.

AV 9-12


Internet Services Configuration Files
/usr/internet/ip/132.147.209.180
HOSTNAME
ns_httpd
publish
sco_ftp
sco_mail
bin
dev
etc
lib
pub
usr
incoming
NADM0912
© The Santa Cruz Operation, Inc.

Exercise: Configuring anonymous FTP

Step 1.	Configure your server (the *net0* interface) to allow anonymous users to download files using the Steps in Section 9.3.2 on page 9-29.
Step 2.	View the files and directories that are created in */usr/internet/ip/a.b.c.d* for this networking interface.
Step 3.	Create a dummy file in the *pub* subdirectory.
Step 4.	Use another system to connect to this server using **ftp(ADMN)**. Log in as user **ftp** and use an e-mail address as your password (for example, `joe@aol.com`). Refer to the ***Network Commands Job Aid*** for help using **ftp**.
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.

9.3.3 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.

9.3.3.1 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:

Real A user is considered *real* when the account name and password provided at authentication time exist in the */etc/passwd* and */etc/shadow* files, and the user is not a member of a guest group.

Guest A user is considered a *guest* when the account name provided at authentication time is a member of a guest group.

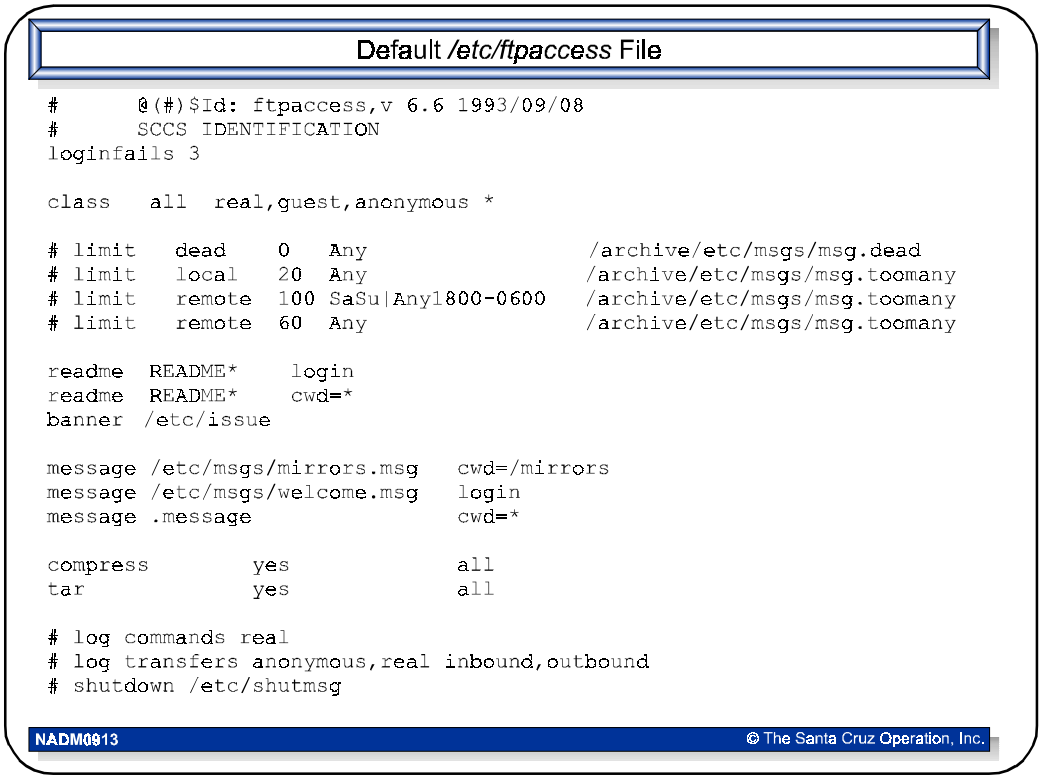
Anonymous A user is considered *anonymous* when the account name provided at authentication time is either **ftp** or **anonymous**.

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 anyone trying to access your host using FTP.

To limit how people can access your host with FTP, create user classes and class limits. Use **vi**(C) to edit the *ftpaccess*(SFF) file.

AV 9-13



Instructor Guide

9.3.3.1.1 The *ftpaccess* file

AV 9-13: Default */etc/ftpaccess* File
Use this slide to discuss the entries listed below. Have the students view the **ftpaccess**(SFF) manual page for further clarification of these entries.

Most FTP server configuration is done using the */etc/ftpaccess* file, described in detail in the *ftpaccess*(SFF) manual page. AV 9-13 shows the *ftpaccess* file that is installed by default. It contains these entries:

`loginfails`	Allows users to log into your ftp server three consecutive times unsuccessfully before they are forced to disconnect and try again.
`class`	Defines a class of users called *all* 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.
`limit`	Limits access by `class` to `n` simultaneous users at the specified time and dates. The message file specified is displayed when a user is denied access.

Illustrate an example on the white board. Include a limit statement as well.

`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 9-13 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*.

Point out that if the asterisk (*) is included, all files beginning with README are displayed. Also, pathnames can be included in `readme` entries. For anonymous users, the path must be relative to the base of the anonymous FTP directory tree.

`message`	Similar to the `readme` entry, except that the actual contents of the file are displayed. You would typically include notices of new entries or changes to the directory structure in these files. Keep them short to cut down on Internet traffic.
`compress`	Transparently compresses files before transfer. See Section 9.3.3.3, "FTP conversions and the ftpconv file," on page 9-37 for details.

This can significantly reduce Internet traffic, download times, etc.

`tar`	Allows a user to get an entire directory from the server with one command. See Section 9.3.3.3, "FTP conversions and the ftpconv file," on page 9-37.
`log`	Enables logging of file transfers based on type of transfer or user.
`shutdown`	The server checks for a planned system shutdown and denies access within a specified time prior to shutdown.

There are several other entries that can be specified in this file. Those most useful for security include: `passwd-check, chmod, delete, overwrite, log commands`, and `log transfers`. Another useful entry is `banner`, but check the manual page before using.

Exercise: Modifying *ftpaccess*(SFF)

Modify the */etc/ftpaccess* 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.

Go through the manual page for *ftpaccess*(SFF) to give students an idea of what else is available in this file that has not been discussed.

9.3.3.2 Denying access to FTP by username or shell

You can prevent certain account names from accessing the FTP server by listing them in the */etc/ftpusers* file. As stated in the manual page for **ftpd**(ADMN), account names allowed to access FTP:

- Must exist on the system, or be either **anonymous** or **ftp** if anonymous FTP is configured
- Cannot have a null password
- Must not be listed in the */etc/ftpusers* file, unless it is **anonymous** or **ftp**
- Must have a login shell that exists in the */etc/shells* file

The */etc/shells* file lists one shell entry per line.

Exercise: Viewing the */etc/shells* file

Use the **cat**(C) command to view the default entries in the */etc/shells* file.

9.3.3.3 FTP conversions and the *ftpconv* 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/ftpconv* file which is documented in the *ftpconv*(SFF) manual page. Modify this file if you need to add conversions not present by default. The conversions present by default include:

NOTE: The word ZIP is used in all capitals to encompass all forms of zip commands, from Info-Zip's **zip** command to the GNU **gzip** command. The commands were not explicitly listed because none of them exist on SCO OpenServer Release 5 machines by default. Info-Zip's package must be purchased separately. GNU **gzip** can be obtained from the SkunkWare distribution.

- Compression with **compress**(C), **gzip**, and **zip**
- Archiving directories using **tar**(C) and **cpio**(C)
- 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**(C)) 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**(C) 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.

9.3.3.4 The */etc/ftpgroups* file

The SCO OpenServer Release 5 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 *ftpaccess*(SFF) file to */etc/ftpgroup*.

Step 2. Create the */etc/ftpgroup* 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**(C) to create the entry in the */etc/ftpgroup* file, minus the password field.

To obtain an encrypted password string that is added to */etc/ftpgroup*, use the **passwd**(C) 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/ftpgroup* file.

See Instructor's note on the next page.

Step 3. Change the group ownership to the **sys** group for files you only want members of the **privgroup** pseudo-group to access.

For example, to copy **root**'s password:

Step 1. Enter **vi /etc/shadow /etc/ftpgroups**.

Step 2. Place the cursor on the first character of root's encrypted password.

Step 3. Yank **root**'s password into the buffer by typing **`yt:`** without pressing ENTER.

Step 4. Switch to the other file by entering **:n** ENTER

Step 5. Position the cursor at the first colon in the entry.

Step 6. Paste the contents of buffer a by typing **p**.

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**.

9.3.4 Flags for ftpd(ADMN)

There are many flags for **ftpd** that can be of use to a network administrator. Logging sessions and writing debugging information are just two types of flags that can be set. Refer to the manual page for **ftpd**(ADMN) for more information.

Exercise: Modifying the existing ftpd entry

Step 1.	Read the manual page for **ftpd**(ADMN) to determine which command-line flags cause it to log all commands received from remote FTP clients to the *syslog* file.
Step 2.	Modify the */etc/inetd.conf* file to cause **ftpd** to log all FTP sessions to the *syslog* file.
Step 3.	Stop and restart the **inetd** service.
Step 4.	Verify that **ftpd** is now logging.

Instructor Guide

Walk the students through each of the flags in the indicated manual page. All of the flags should now make sense to the students.

9.4 Configuring Time Synchronization

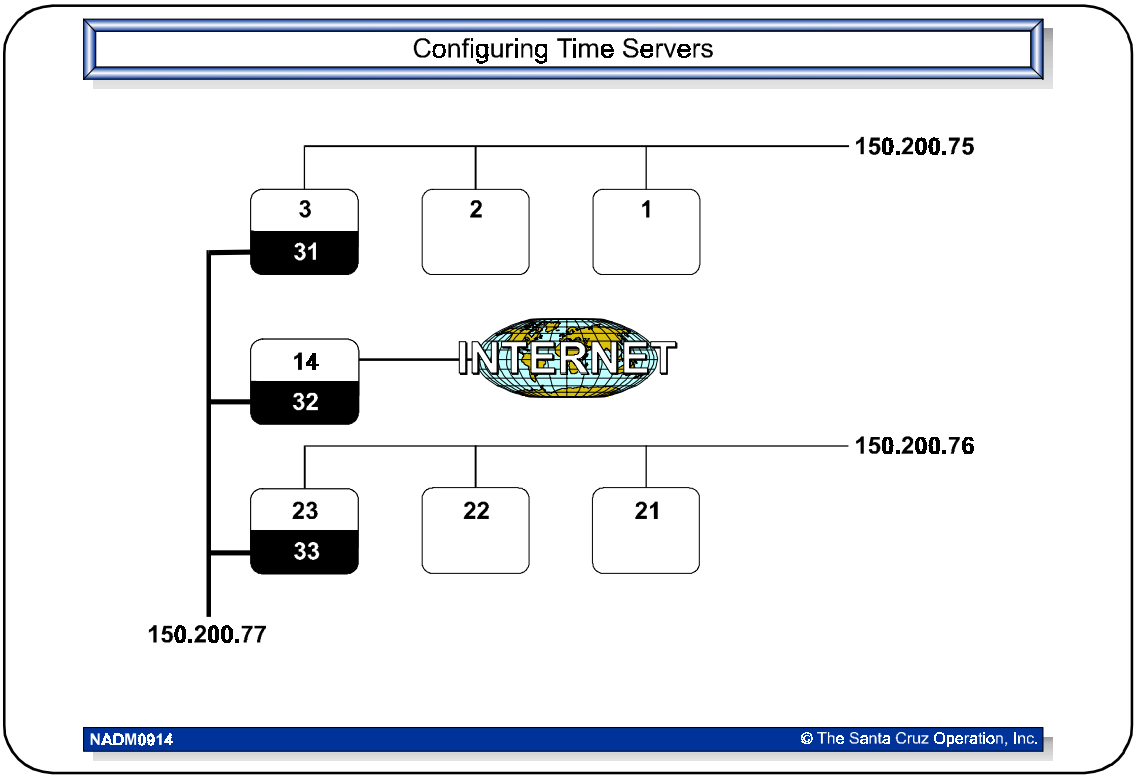
Provide examples of why it is useful (if not important) that cooperating hosts agree on the current date and time. Include NFS, time-stamped logs on database servers, etc.

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.

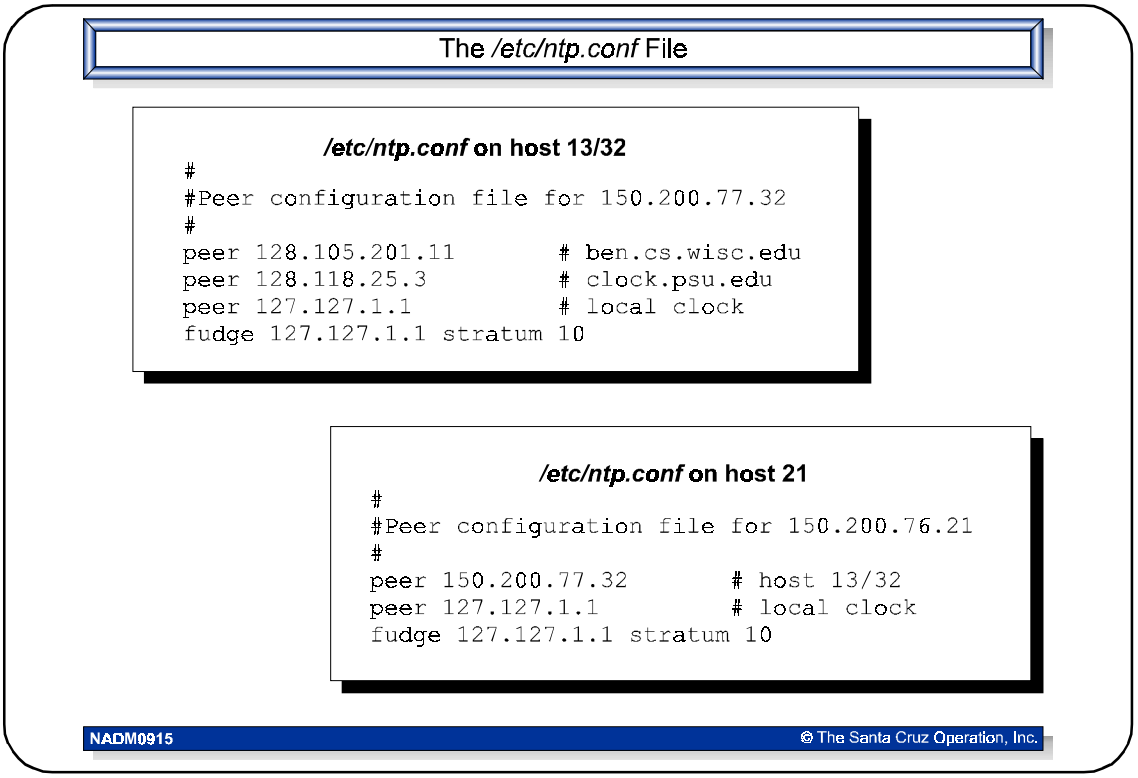
The **xntpd**(ADMN) daemon provides an easy way to have your hosts agree on the current date and time. The **xntpd** daemon implements the Network Time Protocol (NTP). With NTP:

- 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.

AV 9-14


Configuring Time Servers
150.200.75
3
31
2
1
14
32
INTERNET
150.200.76
23
33
22
21
150.200.77
NADM0914
© The Santa Cruz Operation, Inc.

AV 9-15


The /etc/ntp.conf File
/etc/ntp.conf on host 13/32
#
#Peer configuration file for 150.200.77.32
#
peer 128.105.201.11 # ben.cs.wisc.edu
peer 128.118.25.3 # clock.psu.edu
peer 127.127.1.1 # local clock
fudge 127.127.1.1 stratum 10
/etc/ntp.conf on host 21
#
#Peer configuration file for 150.200.76.21
#
peer 150.200.77.32 # host 13/32
peer 127.127.1.1 # local clock
fudge 127.127.1.1 stratum 10
NADM0915
© The Santa Cruz Operation, Inc.

AV 9-14: Configuring Time Servers
Use this slide to illustrate which hosts should be configured to poll which time servers.

For example, for the intranet shown in AV 9-14, configure:

- Host 13/32 to poll a primary and secondary server
- All other hosts to poll host 13/32

To configure a host to run **xntpd**(ADMN):

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/ntp.conf* file on each host:

- Add an entry for each time server
- Add the local clock at address 127.127.1.1.
- Add a line that sets the stratum of the local clock to 10. Do this by adding the following line containing the `fudge` keyword after the entry for the local clock:

```
fudge 127.127.1.1 stratum 10
```

Step 4. Start the NTP daemon:

```
xntpd
```

The presence of the */etc/ntp.conf* file will cause **xntpd** to start at system boot.

Step 5. Use the **ntpq** and **xntpdc** commands to list servers this host is currently polling.

AV 9-15: The */etc/ntp.conf* File
The slide shows */etc/ntp.conf* for host 13/32 configured to poll from a primary (*clock.llnl.gov*) and secondary (*clepsydra.dec.com*). The local host is also added and set to stratum 10.

9.5 Lesson Summary

This lesson has discussed adding virtual domains and configuring Internet services for networking interfaces. Additional descriptions of HTTP, FTP, and NTP protocol configuration were presented.

Review questions

1. Do all TCP/IP-based daemons support virtual domains?

2. What is HTTP and how does it relate to the World Wide Web?

3. What are two ways to access the Netscape **Server Administration** interface, assuming you are properly authorized?

4. What is the default directory used for transferring files (in)to an anonymous FTP server?

5. What is the primary FTP server configuration file?

6. When you use **ftp** to access an account on a remote machine without a password, will it always fail? Explain.

7. Which file contains configuration information for the **xntpd**(ADMN) daemon?

9.6 Lab: Configuring your host to use NTP

In preparation for using time-sensitive client/server programs, configure your host to synchronize its clock to that of the instructor's host.

Procedure

To configure your host to run **xntpd**(ADMN):

Step 6. Create the */etc/ntp.conf* file and add:

- The instructor's time server, using the IP address you are given
- The local clock at address 127.127.1.1
- An entry to set the stratum of the local clock to 10:

```
fudge 127.127.1.1 stratum 10
```

Note that **fudge** is a keyword used to configure reference clocks.

Step 7. Start the NTP daemon:

```
xntpd
```

Step 8. Use the **ntpq** and **xntpdc** commands to list the server this host is currently polling.

Step 9. Monitor the time on your and the instructor's hosts to verify they converge. This will not happen immediately—check it periodically over the next hour or so.

9.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

compress(C)	Compresses files. Used in *ftpconv*(SFF)
cpio(C)	Copy-in-out archiving command. Used in *ftpconv*(SFF)
ftp(C)	Internet file transfer program
ftpd(C)	Internet File Transfer Protocol server
ntpq(C)	Standard NTP query program
scoadmin(ADM)	Interface-independent program that provides access to the various managers
scohelp(XC)	Provides access to web-based help and manual pages
sendmail(ADM)	Server daemon for e-mail (an MTA)
start-admin	Starts the Netscape Administration Server
stop-admin	Starts the Netscape Administration Server
tar(C)	Creates tape archives (used in *ftpconv*(SFF))
xntpd(ADMN)	NTP protocol daemon
xntpdc(ADMN)	Special NTP query program

Relevant files

/etc/ftpaccess	Contains parameters that control most of the configurable behavior of **in.ftpd**(ADMN). Documented in *ftpaccess*(SFF).
/etc/ftpconv	Contains rules for converting (compressing and archiving) files during FTP sessions. Documented in *ftpconv*(SFF).

/etc/ftpgroups	Contains a list of pseudo-group to real-group mappings for use with the SITE extended FTP commands. Documented in *ftpgroups*(SFF).
/etc/ftpusers	Contains a list of users which cannot be used to authenticate an FTP session. Documented in *ftpusers*(SFF).
/etc/group	Contains a list of real groups. Documented in *group*(SFF).
/etc/inetd.conf	Contains configuration information for **inetd**(ADMN). Documented in *inetd.conf*(SFF).
/etc/ntp.conf	Contains the configuration information for the **xntpd**(ADMN) daemon
/etc/passwd	Contains a list of real users and their home directories. Documented in *passwd*(SFF).
/etc/services	Lists names of server daemons with their protocol and port number. Documented in *services*(SFF).
/etc/shadow	Contains a list of real users and their encrypted passwords. Documented in *shadow*(SFF).
/etc/shells	Contains a list of login-shells a real user must have listed in the last field of their *passwd*(SFF) entry in order for authentication to succeed
/usr/adm/syslog	Log file for **syslogd**(ADM)
/usr/internet	Default directory for system-wide Internet services configuration files.
/usr/internet/ip/a.b.c.d	Default directory for Internet services configuration files. If virtual domains have not been configured, the directory is */usr/internet/ip/0.0.0.0/publish/htdocs.*
/usr/internet/ip/a.b.c.d/sco_ftp	Default home directory of the **ftp** user
/usr/internet/ns_httpd	Default home directory for system-wide Netscape FastTrack server configuration files
/usr/internet/ip/a.b.c.d/ns_httpd/htdocs	Default directory for HTML pages served by the Netscape FastTrack server for the interface with IP address *a.b.c.d.*

/usr/internet/ip/a.b.c.d/ns_httpd/config
: Default directory for files configuring the behavior of the Netscape FastTrack server running on port 80

/usr/internet/ip/a.b.c.d/ns_httpd/logs
: Default directory for log files for the Netscape FastTrack Server running on port 80

/usr/ns-home/start-admin
: Netscape Administration Server daemon startup program

/usr/ns-home/stop-admin
: Netscape Administration Server daemon shutdown program

9.8 Answers to Review Questions

This section provides suggested answers to review questions.

1. No, support of virtual domains is up to the individual daemons.

2. HTTP is the HyperText Transport Protocol and it is the language spoken by all web servers on the World Wide Web.

3. The Netscape **Server Administrator** can be run several ways:

   - On the local host, run **scoadmin ⇨ Netscape ⇨ Netscape FastTrack Server**.

   - On the local host, start the Netscape Communicator and load URL *http://local-hostname:620*.

   - On a remote host, start a web browser and load URL *http://remote-hostname:620*.

4. When writable anonymous FTP transfers are allowed, read, write, and execute permission are given to */usr/internet/ip/a.b.c.d/sco_ftp/pub/incoming*. This is the only directory that has write permission by default.

5. Most FTP configuration information is found in the */etc/ftpaccess* file.

6. Yes. The **ftpd** daemon will not allow access to accounts without passwords. See Section 9.3.3.2, “Denying access to FTP by username or shell,” on page 9-37.

7. The */etc/ntp.conf* file contains the configuration information for the **xntpd**(ADMN) daemon.

10

Network Security

Instructor Guide

Lesson 10

Network Security

Suggested lesson duration

This lesson should take approximately 2 hours to complete, including the lab exercise.

Pre-lesson classroom preparation

You might want to set up a machine running a packet sniffer and demonstrate the security risk they present. You can also monitor the effect of packet filters. The Skunkware 97 CD-ROM contains a packet sniffer for SCO OpenServer Release 5.

Have the Optional Services CD-ROM available to install the SCO Internet Security Package and the Netscape Proxy Server.

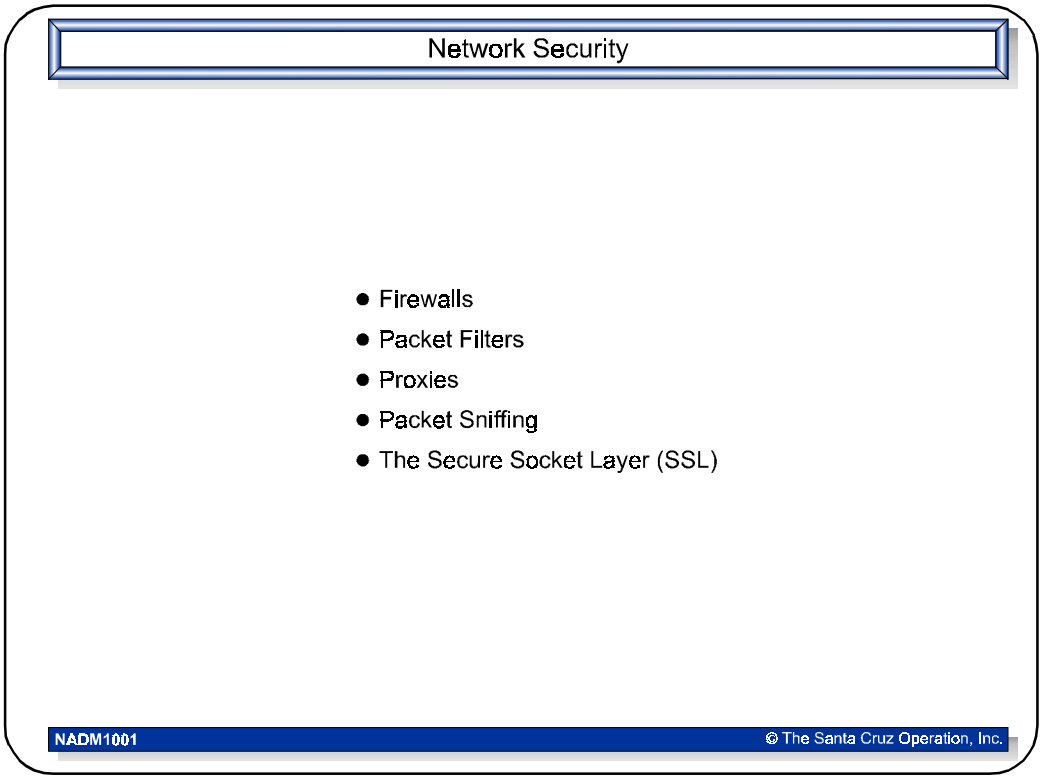
Instructor preparation

You must be able to configure packet filters for SCO PPP. You must also be familiar with the optional SCO Internet Security Package and the Netscape Proxy Server.

Good references for security concepts include the following books and SCO documentation:

- **scohelp ⇨ Networking Guide ⇨ Configuring Point-to-Point Protocol (PPP) ⇨ Configuring packet filters**
- **scohelp ⇨ Internet Family Documentation Guide ⇨ Netscape Proxy Server Administrator's Guide** and **SCO Internet Security Package User's Guide.**
- Cheswick, William R. and Bellovin, Steven M., ***Firewalls and Internet Security***, Reading, MA: Addison-Wesley Publishing Company, 1994, ISBN: 0-201-63357-4
- Chapman, D. Brent, and Zwickey, Elizabeth D., ***Building Internet Firewalls***, Sebastopol, CA: O'Reilly & Associates, 1995, ISBN: 1-56592-124-0
- Garfinkel, Simson and Spafford, Gene, ***Practical UNIX & Internet Security***, Sebastopol, CA: O'Reilly & Associates, 1996, ISBN: 1-56592-148-8

AV 10-1



Instructor Guide

Introduction

Earlier lessons described how to connect your machine to a local area network and the Internet, and how to provide services to the network. These connections provide access to a variety of information for you and for those interested in the services you have to offer. 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 obstruct those who intend to damage or misuse your system, or the data it contains.

Objectives

This lesson discusses network security features and describes how to configure those built into SCO OpenServer Release 5. By the end of this lesson, you will be able to:

- Discuss the use of firewalls
- Configure packet filters to restrict the flow of packets through your system
- Describe the function of proxy servers
- Discuss the dangers and uses of packet sniffers and how to protect against them by using the Secure Socket Layer (SSL)

AV 10-1: Network Security
This first slide describes the objectives covered in this lesson.

10.1 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 SCO OpenServer Release 5 system can accomplish the above. To achieve this, the system should be:

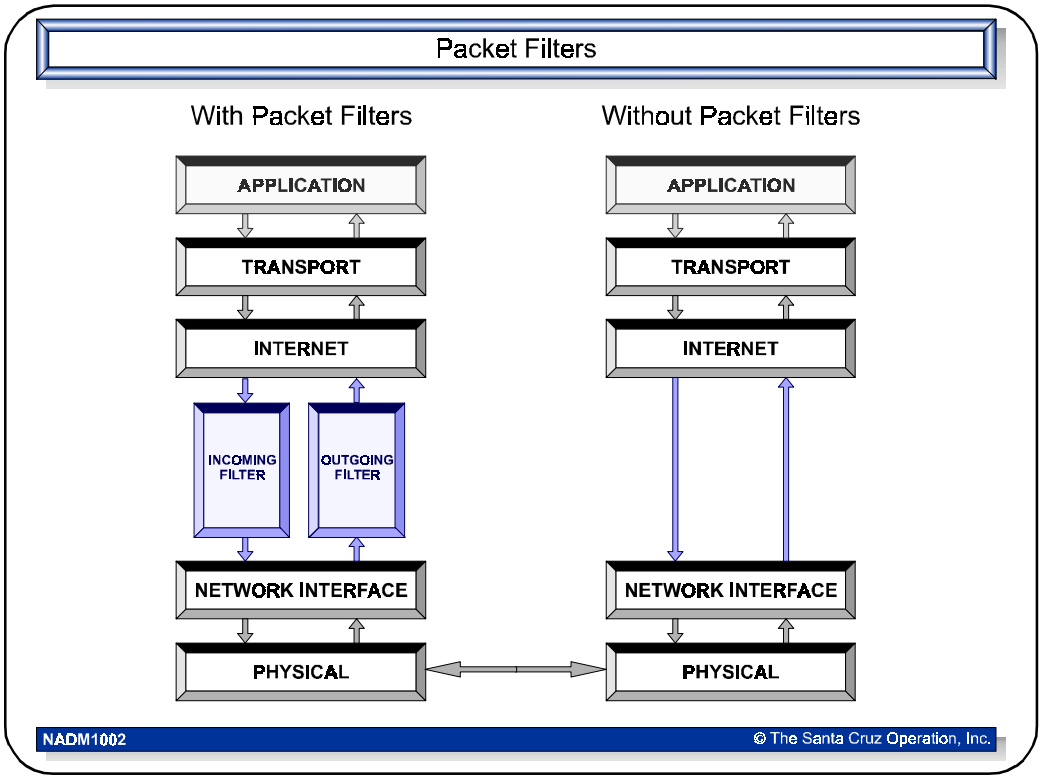
- Set up as the connection between the Internet and the local networks
- Configured with security features such as *packet filters* and *proxy servers*
- Running minimal services and have as few accounts as possible to reduce risk of penetration

Instructor Guide

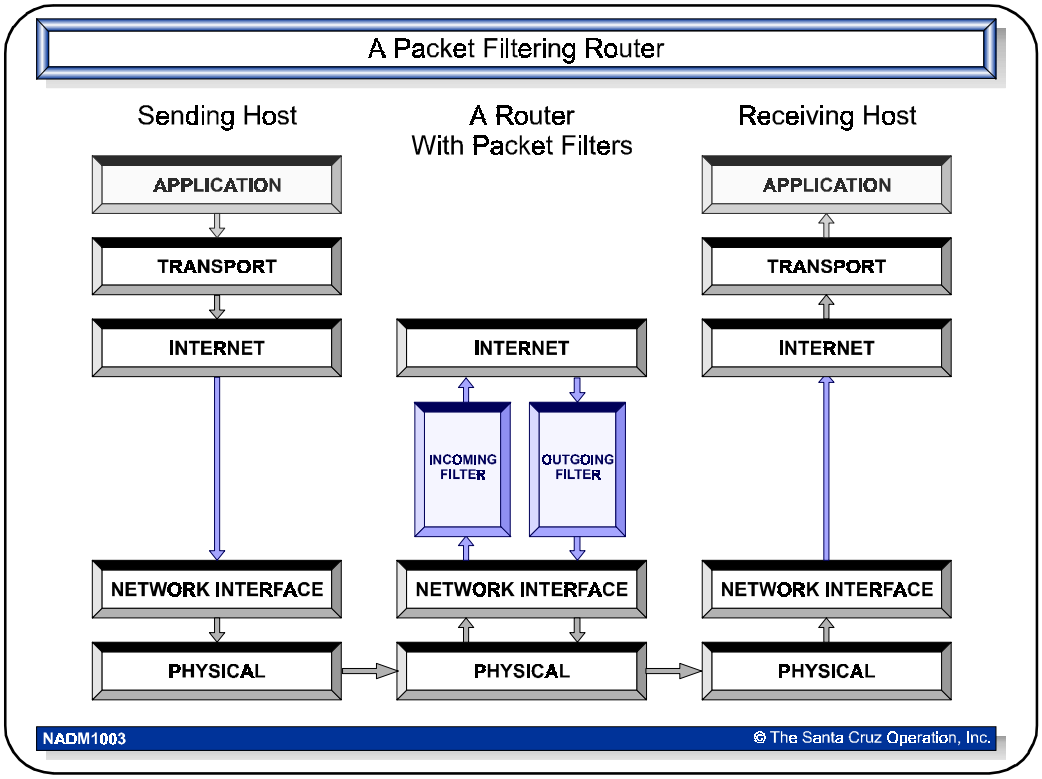
Discuss how students can reduce services running on the system by editing the */etc/inetd.conf* file.

A firewall is not a simple switch that you turn 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 10-2



AV 10-3



Instructor Guide

10.2 Packet filters

AV 10-2: Packet Filters
This diagram shows where packet filters fit into the network layer model. Emphasize the differences between this and the operation of TCP wrappers. Also, point out why these affect packets routed where as TCP wrappers do not.

AV 10-3: A Packet Filtering Router
This diagram shows a packet traveling from a system on the left though a packet filtering router and on to the destination system on the right.

A packet filter is a set of rules that control the delivery of packets between the network and Internet layers of the protocol stack.

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 SCO OpenServer systems, packet filters can be configured on any WAN interface that uses the PPP serial line protocol. With the addition of the optional SCO Internet Security Package, dynamic packet filters can be configured on any networking interface.

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

10.2.1 Static packet filters

Static packet filters can be created for PPP interfaces on a per-endpoint basis to control packets in each direction:

Incoming	Restricts which packets get into the machine from the incoming PPP interface
Outgoing	Restricts which packets get out of the machine through the outgoing PPP interface

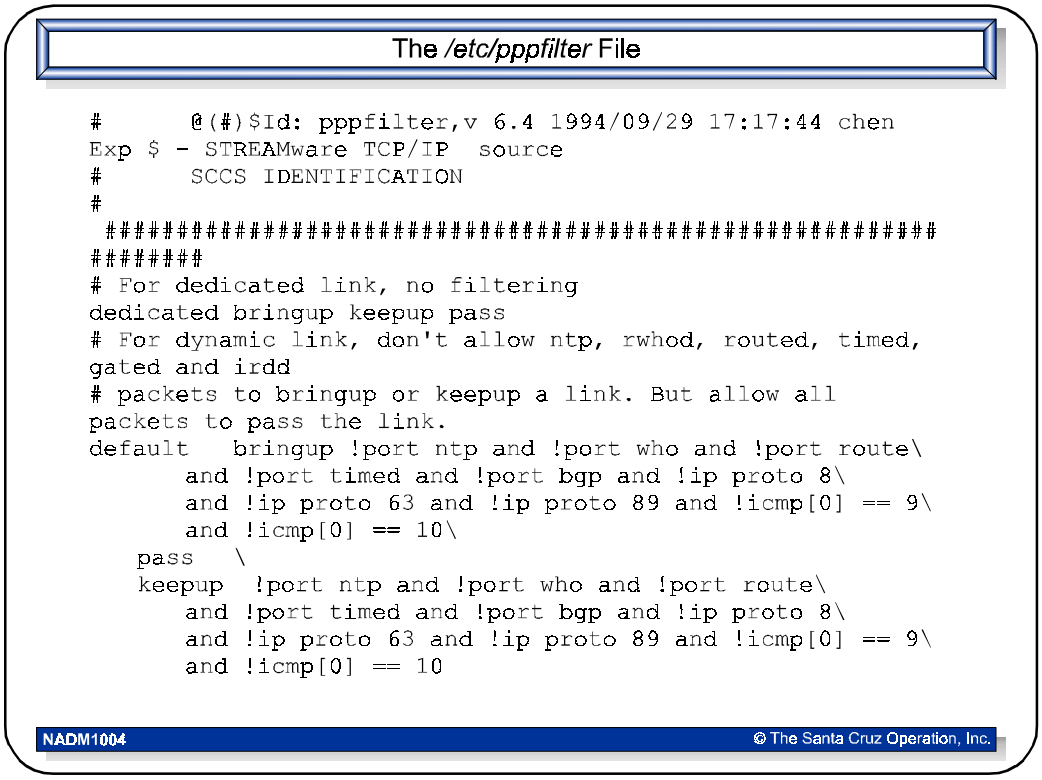
You can also specify which packets are qualified or disqualified to pass through the interface based on:

- Transport protocol type (TCP, UDP, ICMP, for example)
- IP destination or source field
- Destination or source port field in IP/TCP or IP/UCP packets
- Packet length
- IP packet type (that is, broadcast or multicast)

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).

Point out why this makes filters more secure.

AV 10-4


The /etc/pppfilter File
@(#)$Id: pppfilter,v 6.4 1994/09/29 17:17:44 chen Exp $ - STREAMware TCP/IP source
SCCS IDENTIFICATION
#
##
########
For dedicated link, no filtering
dedicated bringup keepup pass
For dynamic link, don't allow ntp, rwhod, routed, timed, gated and irdd
packets to bringup or keepup a link. But allow all packets to pass the link.
default bringup !port ntp and !port who and !port route\
and !port timed and !port bgp and !ip proto 8\
and !ip proto 63 and !ip proto 89 and !icmp[0] == 9\
and !icmp[0] == 10\
pass \
keepup !port ntp and !port who and !port route\
and !port timed and !port bgp and !ip proto 8\
and !ip proto 63 and !ip proto 89 and !icmp[0] == 9\
and !icmp[0] == 10
NADM1004
© The Santa Cruz Operation, Inc.

10.2.2 Creating a PPP packet filter

Details of configuring PPP packet filters are based on SCO PPP in this section. Packet filters for Morning Star PPP are not described in detail. Refer students to the documentation for more information.

Packet filters for SCO PPP interfaces are configured by editing the PPP configuration file, */etc/pppfilter*, which has this format:

```
tag   keyword expression [ keyword expression ] [ keyword expression ]
```

where:

tag — Specifies the interface to which this filter applies. The */etc/ppphosts* file has a corresponding `filter=tag` entry that is an index into */etc/pppfilter.*

keyword — Can be one of these values:

- `bringup`—the expression that follows determines which packets allow the interface to be brought up
- `keepup`—the expression that follows determines which packets will reset the idle time (which allows the interface to stay up)
- `pass`—the expression that follows determines which packet is allowed to traverse the PPP interface

expression — Can be one or more primitives (see *packetfilter*(SFF) for a complete list) or a combination of primitives. These expressions determine the criteria for packets to be allowed through the interface.

The default filter for all PPP interfaces allows any packet to pass. However, only certain packets will bring up outgoing (dynamic) links. This is illustrated in AV 10-4.

AV 10-4: The */etc/pppfilter File*
This slide shows the default */etc/pppfilter* file for SCO PPP. Use this to explain how expressions are used to define which packets to filter.

A simple example of a filter entry for an incoming interface that blocks **telnet**(TC) access is:

```
in    pass    port !telnet
```

The entry in */etc/ppphosts* referencing this filter would be:

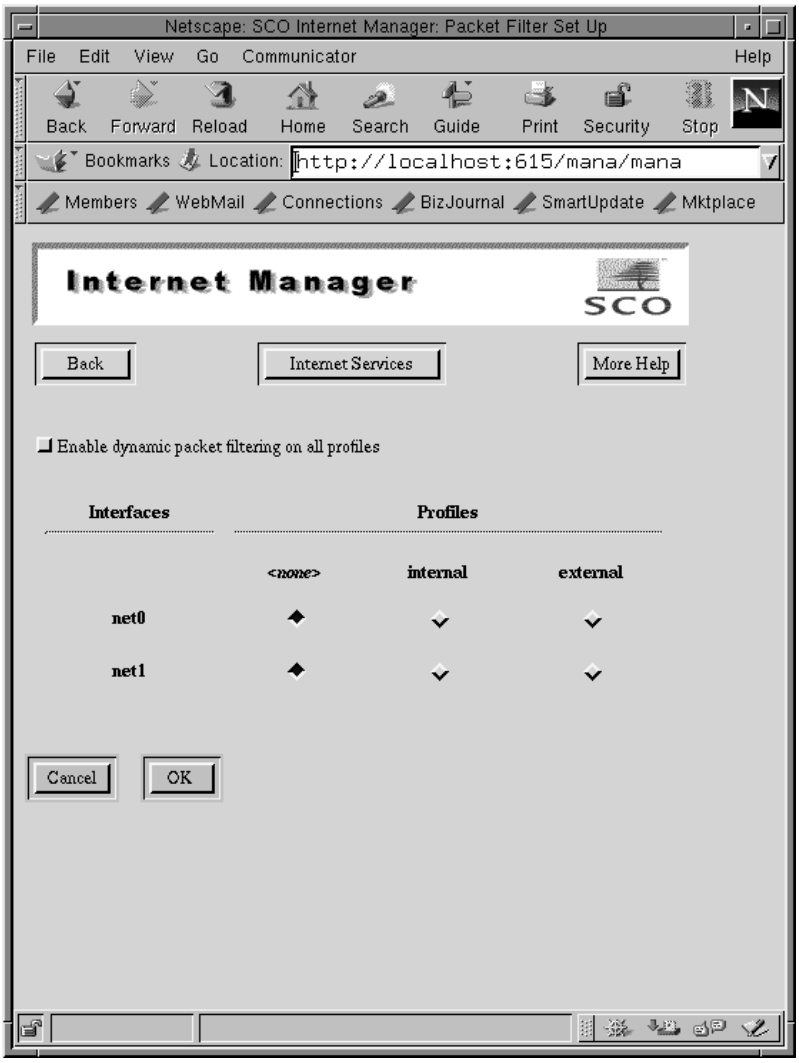
```
*nppp local=150.100.25.1 remote=150.100.25.10 filter=in
```

For more information about SCO PPP packet filters, see *packetfilter*(SFF). The Morning Star PPP filter file is */usr/lib/mstppp/Filter*. See *ppp.Filter*(MST_PPP) for more information.

This would be a good time to do the lab in Section 10.6, “Lab: Implementing a PPP Packet Filter,” on page 10-34.

AV 10-5

Set Up Packet Filter Data Flow



NADM1005 © The Santa Cruz Operation, Inc.

10.2.3 Dynamic packet filters

The Internet Security Package is a separately licensed product that allows you to create dynamic packet filters for any networking interface, not just the PPP interfaces. It can also be configured to encrypt data and network addresses using *virtual private networks* (VPN).

Dynamic firewall technology examines every packet passing through the network interface. When a packet meets the requirements of the packet filtering rules, it triggers a new set of filtering rules to be inserted into the filter process for some period of time.

The ability to adapt to network traffic provides a distinct advantage over ordinary static packet filtering. For example, dynamic filtering allows you to grant access from a particular destination to an application for a specified period of time.

After it is installed, the Internet Security Package is configured using the **Internet Manager**. Basic packet filters can easily be created using the **Internet Manager** graphical interface. Creating more complex filters requires editing the */usr/internet/lib/mst/filters/api* configuration file.

This procedure describes how to install the Internet Security Package and configure a simple filter which prevents inbound **telnet**(TC) access from the Internet.

Step 1. Install the Internet Security Package from the Optional Services CD-ROM using the **Software Manager**.

Follow the prompts to install the 60-day trial version of the product.

Step 2. Start the **Internet Manager**. Enter **admin** as the user ID and the **root** password.

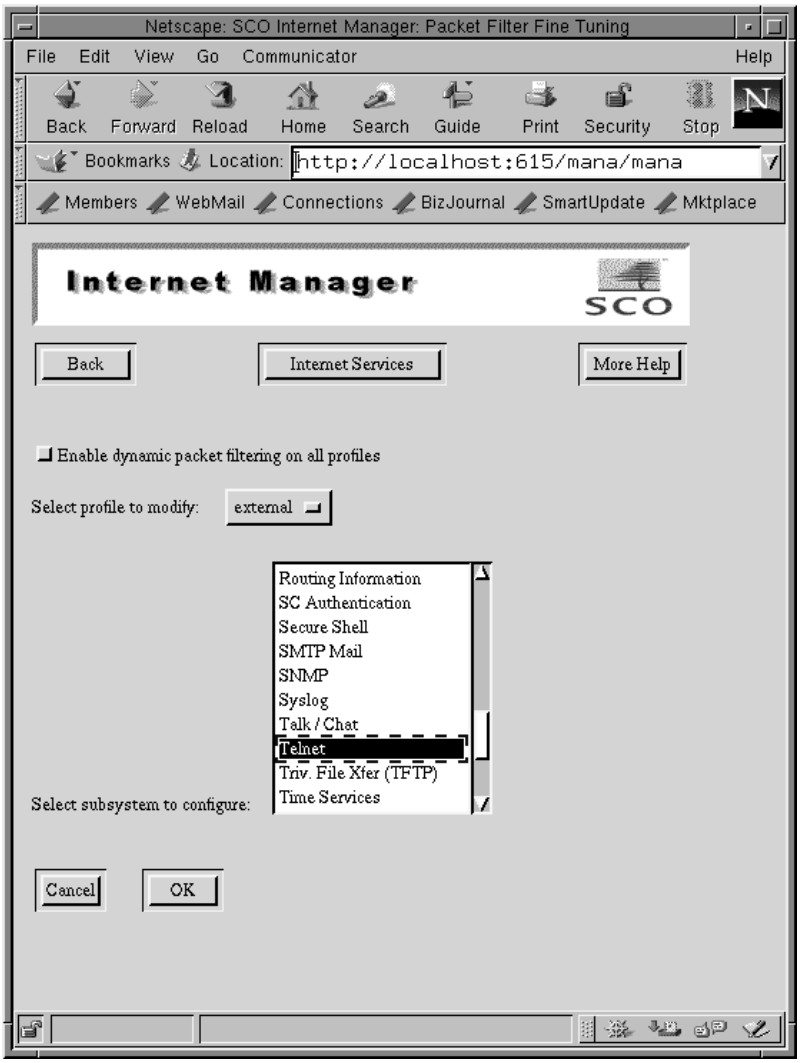
Step 3. Click on the **Security** icon.

Step 4. Click on **Set Up Packet Filter Data Flow**.

AV 10-5: Set Up Packet Filter Data Flow
This slide shows the dynamic packet filter configuration screen accessed from **Internet Manager** ⇨ **Security** ⇨ **Set Up Packet Filter Data Flow**.

AV 10-6

Fine Tune Packet Filter Configuration



NADM1006 © The Santa Cruz Operation, Inc.

Step 5. Select the “external” profile for the networking interface used to access the Internet. This is **`net0`** in the example shown in AV 10-5.

Profiles are used to group interfaces, providing an easy way to apply a set of packet filters to a group of interfaces. Two default profiles are defined:

- internal—used for interfaces to internal networks (intranets)
- external—used for interfaces to external networks (Internet)

In this example setup, external **telnet** access is being prevented so “external” is selected.

Step 6. Toggle “Enable dynamic packet filtering.”

Step 7. Click on **OK**.

Step 8. Click on **Fine Tune Packet Filter Configuration**.

Step 9. Select “external” as the profile to modify.

Step 10. Scroll through the list and select the **telnet** subsystem.

Step 11. Click on **OK**.

Step 12. Toggle “deny” under **Inbound Enable**. This prevents inbound access from the external interfaces.

Step 13. Toggle “allow” under **Outbound Enable**. This allows users to **telnet** from the system out to the Internet.

Step 14. Click on **OK**. For this example, no additional client or server configuration is needed.

Step 15. Click on **Back**, then click on **Done**.

The system will no longer allow **telnet** access on all networking interfaces included in the external profile.

For information about configuring dynamic packet filters, see **scohelp ⇨ SCO Internet Family Documentation ⇨ SCO Internet Security Package User’s Guide**.

AV 10-6: Fine Tune Packet Filter Configuration
This slide shows how dynamic packet filters are configured.

Exercise: Creating a telnet packet filter

In this exercise, install the Internet Security Package and configure an inbound **telnet** filter.

Step 1. Access your server from another system on the network using **telnet**. This should succeed.

Step 2. Install the Internet Security Package.

Step 3. Start the **Internet Manager** and access the security configuration screen.

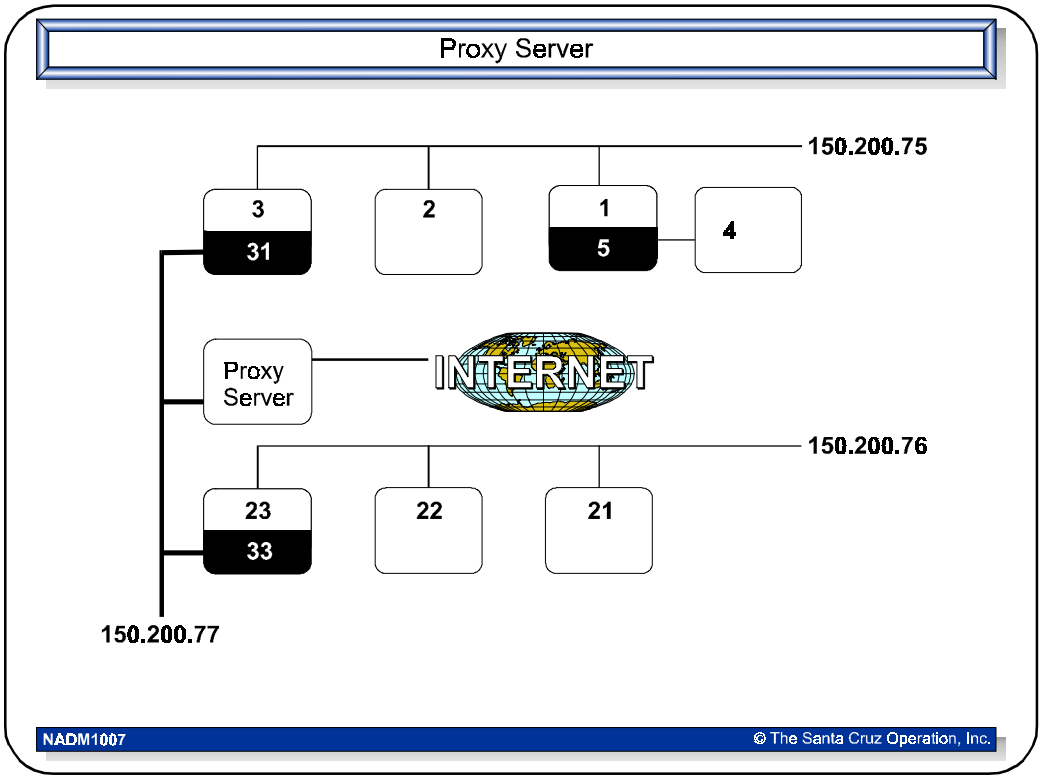
Step 4. Assign your LAN interface to the external profile.

Step 5. Enable dynamic filtering on all interfaces.

Step 6. Configure an inbound **telnet** filter for the external profile.

Step 7. Access your server from another system on the network using **telnet**. This should fail.

AV 10-7



Instructor Guide

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 SCO OpenServer Release 5 for this purpose are:

Netscape Proxy Server™	Available as an add-on product from SCO
Squid	Available on the Skunkware CD-ROM
CERN httpd	Source code freely available on the web. Must be compiled to run on SCO OpenServer Release 5.

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.

AV 10-7: Proxy Server
This slide shows where a proxy server might be configured for the network example used in previous lessons.

10.3.1 The Netscape Proxy Server

The Netscape Proxy Server proxies the HTTP, HTTPS, FTP, and Gopher protocols for properly configured web browsers. It is used to enable access to the World Wide Web from clients that do not have full Internet access. For example, the clients may not have Internet-registered IP addresses, or there may be a packet filter that blocks access to the Internet.

The Netscape Proxy Server software is on the SCO OpenServer Release 5 Optional Services CD-ROM and comes with a 60-day evaluation license. To continue using the Netscape Proxy Server after the evaluation license has expired, you must purchase a license.

Install the Netscape Proxy Server using the **Software Manager**:

Step 1. Before installing the Netscape Proxy Server, make sure DNS is running.

Step 2. Start the **Software Manager** as **root**; select **Software ⇨ Install new**; select **"**From hostname**"** and click **Continue;** select CD-ROM as the media device and continue.

Step 3. Select "Netscape Proxy Server (ver 2.5.0d)" when the list of available software is displayed, and click **Install**.

Step 4. Answer "n" if you do not have a permanent license and ENTER to continue.

After the files have been copied from the CD, you see a message that you must install a Netscape browser to administer this server, using URL *http://localhost:446*. Because the Netscape Communicator browser is installed by default, no further browser installation is necessary.

Step 5. Press ENTER to continue.

Step 6. Wait for the installation to complete and click on **OK**.

The Netscape Proxy Server is started and listens on port 8080.

For more information about the Netscape Proxy Server, see **scohelp ⇨ SCO Internet Family Documentation⇨Netscape Proxy Server Administrator's Guide**.

As an alternative to the Netscape Proxy Server, the free Squid proxy server is on the Skunkware 98 CD-ROM in the *./osr5/net/squid/squid-1.1.14.cpio.gz* file. To install, use the **Software Manger** (custom).

After installing it, edit the */usr/local/squid/etc/squid.conf* file, and make the following changes:

- Change the `http_port` to `8080`

Instructor Guide

- Comment out the `httpd_accel` directive.

Start the server with the command line:

```
/etc/squid start
```

10.3.2 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.

These will not work if you omit the trailing slash in the URLs.

Exercise: Using the Netscape Proxy Server

In this exercise, convert a router into a proxy server:

Step 1. Disable routing on a classroom router with the command:

```
# inconfig ipforwarding 0
```

The router is disabled to demonstrate the purpose of a proxy server. The proxy server itself works on an enabled router. If some of the classroom systems are using IP addresses that are not Internet-routable, you can use them for the browser systems and not disable the router for this exercise.

Step 2. Try to access a web server on the other side of the router. This should fail.

Step 3. Install the Netscape Proxy Server on the disabled router.

Step 4. Configure the web browser to use the proxy server.

Step 5. Try again to access a web server on the other side of the router. This time, you should succeed.

Some browsers still fail unless they have access to a DNS server. You might need to configure the client system to query a DNS server that is on its side of the router.

Step 6. Enable the router with the command:

```
# inconfig ipforwarding 1
```

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.

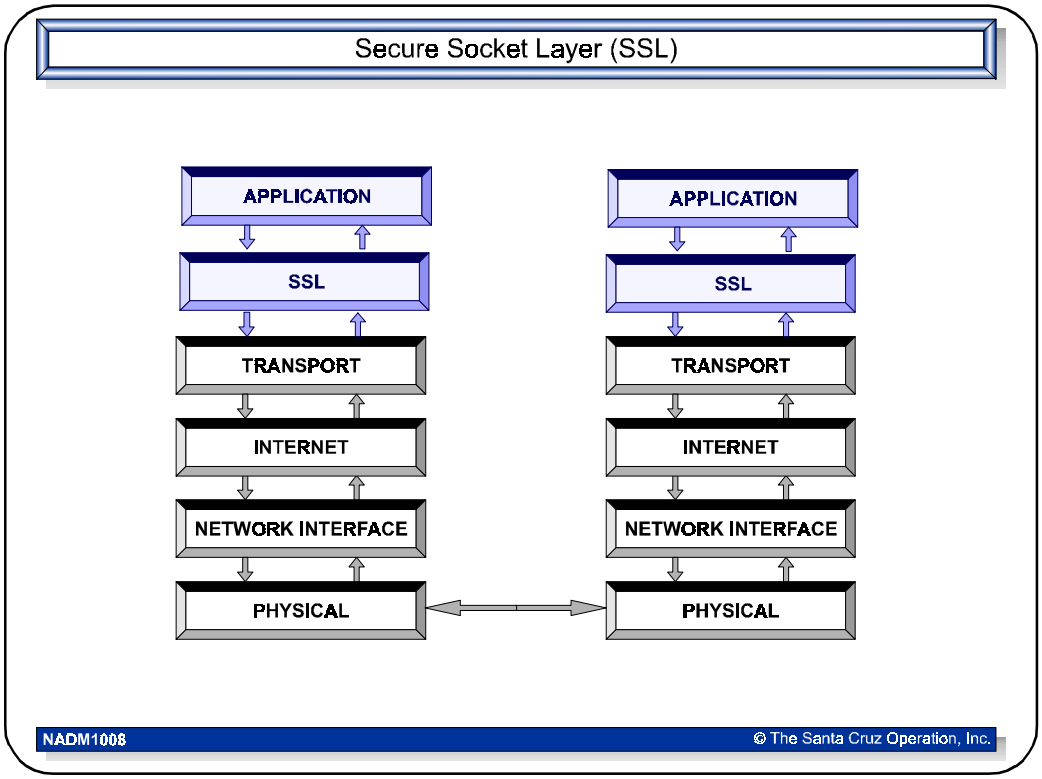
Despite these dangers, packet sniffers can also have legitimate uses:

- Monitoring network traffic for security purposes
- Diagnosis of network problems
- Measuring network traffic

If you have a packet sniffer running, have the students watch a **telnet** session to see how easy it is to snoop passwords sent by **telnet**.

A packet sniffer for SCO OpenServer Release 5 called **tcpdump** can be found on the SkunkWare 97 CD-ROM in the */osr5/net/nettools/tcpdump-3.0.4+* file.

AV 10-8


Secure Socket Layer (SSL)
APPLICATION
SSL
TRANSPORT
INTERNET
NETWORK INTERFACE
PHYSICAL
APPLICATION
SSL
TRANSPORT
INTERNET
NETWORK INTERFACE
PHYSICAL
NADM1008
© The Santa Cruz Operation, Inc.

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 10-8.
- 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 ⇨ SCO Internet Family Documentation⇨Netscape FastTrack Server Administrator’s Guide ⇨ Chapter 7 Encryption and SSL**.

AV 10-8: The Secure Socket Layer
This slide illustrates how data is sent by applications through the SSL and then to the Transport layer (TCP).

The SSL protocol includes support for compressing data before encryption. It also uses different ports for SSL connections than for the same standard service connection. For example:

- https 443/tcp
- ssmtp 465/tcp
- snews 563/tcp
- ssl-ldap 636/tcp
- spop3 995/tcp

The SSL protocol uses two kinds of encryption technologies:

- Symmetric Key Encryption—Uses the same secret key for encryption and decryption. It is quick and efficient but has a problem regarding the secure exchange of keys.
- Asymmetric Key Encryption—Uses different keys for encryption and decryption. This allows the encryption key to be sent out in public and the decryption key to be kept private. Note that this method requires longer key lengths and more processing time encrypting and decrypting.

To maximize efficiency and security, an SSL connection uses both technologies. The server sends the client its public key. The client generates a random number and sends it back encrypted using the public key. This random number is then used as the key to encrypt further communication using symmetric key encryption.

US export regulations restrict the strength of encryption technology that can be exported. For exported products, this limits the length of the keys used in:

- Asymmetric (public) key encryption to 512 bytes
- Symmetric key encryption to 40 bytes

For a comparison of these two restrictions, to date, 512-byte public key encryption has not been broken. A 40-bit symmetric key encryption was broken in one instance in January, 1997 by a network of workstations in 3.5 hours.

This discrepancy between the key strengths does make sense because the same public key is used repeatedly to set up sessions. Symmetric keys are used only for a single session.

SSL uses different keys for integrity, encryption, and authentication. Because export restrictions only regulate key length for encryption, this allows SSL on export products to use stronger keys for integrity and authentication.

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, the Netscape Proxy Server, and the Secure Sockets Layer protocol were also described.

Review questions

1. What qualifies a machine as a firewall?

2. What are packet filters used for?

3. Why would you run a proxy server on a firewall?

4. Can all hosts on the Internet sniff packets sent or received by your system? Explain your answer.

5. What measures can you implement to guard against your packets being sniffed?

10.6 Lab: Implementing a PPP Packet Filter

The goal of this lab is to implement a packet filter on a PPP interface.

Overview

This lab can be done with pairs of systems, one with incoming PPP configured and another with outgoing PPP configured. After you have verified that you can connect to another system using PPP:

1. Bring down your LAN interface to make sure that the WAN interface is the one being used to connect to the remote host.
2. Implement a packet filter on your incoming PPP interface to prevent **ftp**(TC) access from all hosts.
3. Test your filter from the remote host.
4. Disable your filter when you are finished.
5. Restore your LAN interface.

Instructor Guide

If you have an SCO OpenServer Release 5 system configured as a router, you can perform this lab on the router and demonstrate that it blocks packets that are not addressed to the router.

Procedure

To configure the **ftp** packet filter:

Step 1. Mark the LAN interface as *down*:

```
ifconfig net0 down
```

Step 2. Edit */etc/pppfilter*.

Remove or comment out any existing entries.

Add an entry to disallow **ftp** packets. (See Section 10.2.2, “Creating a PPP packet filter,” on page 10-13.)

Step 3. Edit */etc/ppphosts* to add the `filter` keyword and tag to the PPP interface.

Step 4. Signal the PPP daemon to re-read configuration files:

```
kill -1 'cat /etc/pppd.pid'
```

Step 5. Test this packet filter by attempting to **ftp** in from a remote host over a PPP connection.

When you are finished, remove your filter so **ftp** can be used in the remainder of this course.

Step 6. Edit */etc/ppphosts* and remove the `filter=tag` entry.

Step 7. Signal the PPP daemon.

Step 8. Test that incoming **ftp** now works from different hosts.

Step 9. Restore the LAN connection:

```
tcp stop

tcp start
```

10.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

finger(TC)	Displays information about local and remote users
ftp(TC)	Invokes the Internet file transfer program
ifconfig(ADMN)	Configures network interface parameters
inconfig(ADMN)	Configures TCP/IP-related kernel parameters
telnet(TC)	Invokes the remote login program

Relevant files

/etc/inetd.conf	Contains configuration information for **inetd**. See *inetd.conf*(SFF).
/etc/pppfilter	Contains packet filter information for SCO PPP. See *packetfilter*(SFF).
/etc/ppphosts	Contains PPP configuration information. See *ppphosts*(SFF)
/usr/lib/mstppp.Filter	Contains packet filter configuration information for PPP from Morning Star. See *ppp.Filter*(MST_PPP).

10.8 Answers to Review Questions

This section provides suggested answers to review questions.

1. 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.

2. Packet filters restrict all packets meeting certain criteria from passing through an interface and therefore can restrict access to an entire network when implemented on a router.

3. 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.

4. 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.

5. 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

Instructor Guide

Lesson 11

Windows Connectivity

Suggested lesson duration

This lesson should take approximately 3 hours to complete. There is not a lab at the end of the lesson.

Pre-lesson classroom preparation

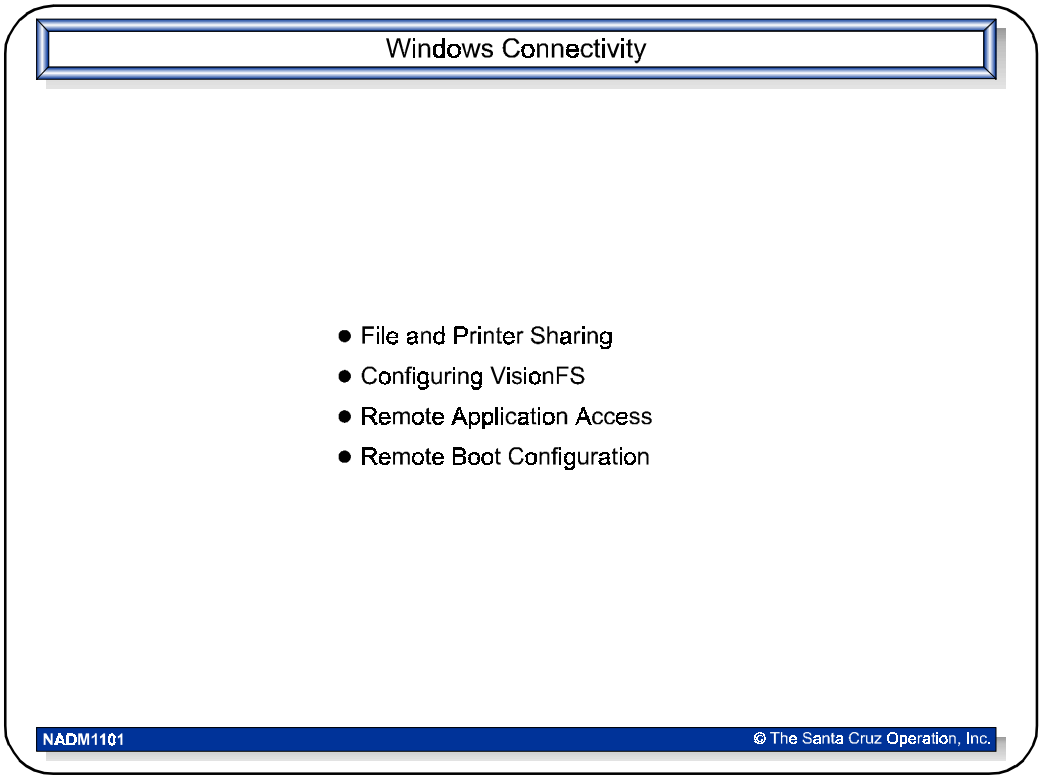
- Have a working classroom network in place.
- Have VisionFS installed on at least one OpenServer system, or have an OpenServer CD-ROM available so that VisionFS can be installed during class.
- Have a printer available if you plan to demonstrate server printer sharing.
- Depending on classroom resources, you have several options for providing Windows access, including:
  - Providing two computers for every student, one running OpenServer and one Windows
  - Providing two computers for every pair of students, one running OpenServer and one Windows
  - Providing one Windows system in the classroom, and changing the exercises into demonstrations
- If you are configuring shared printers in class, have the Windows installation CD available to load the printer driver(s).

Instructor preparation

Configure VisionFS prior to class and familiarize yourself with the product and its use. If you are new to Windows, practice configuring TCP/IP on the version of Windows that is used in class. Also practice using the Network Neighborhood, Windows Explorer, or other file navigation tools.

You might want to refer to ***Introducing VisionFS***, a 120+ page Acrobat document shared by default as *visiontools\docs\vfsintro.pdf*. The default UNIX pathname is */usr/vision/bin/pctools/en_US/docs/vfsintro.pdf*.

AV 11-1



Instructor Guide

Introduction

In addition to communication between UNIX 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, and in some cases the tools, 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, a tool included with SCO OpenServer Release 5, is used to illustrate file and print sharing. This lesson introduces VisionFS in addition to summarizing alternative options for providing similar services.

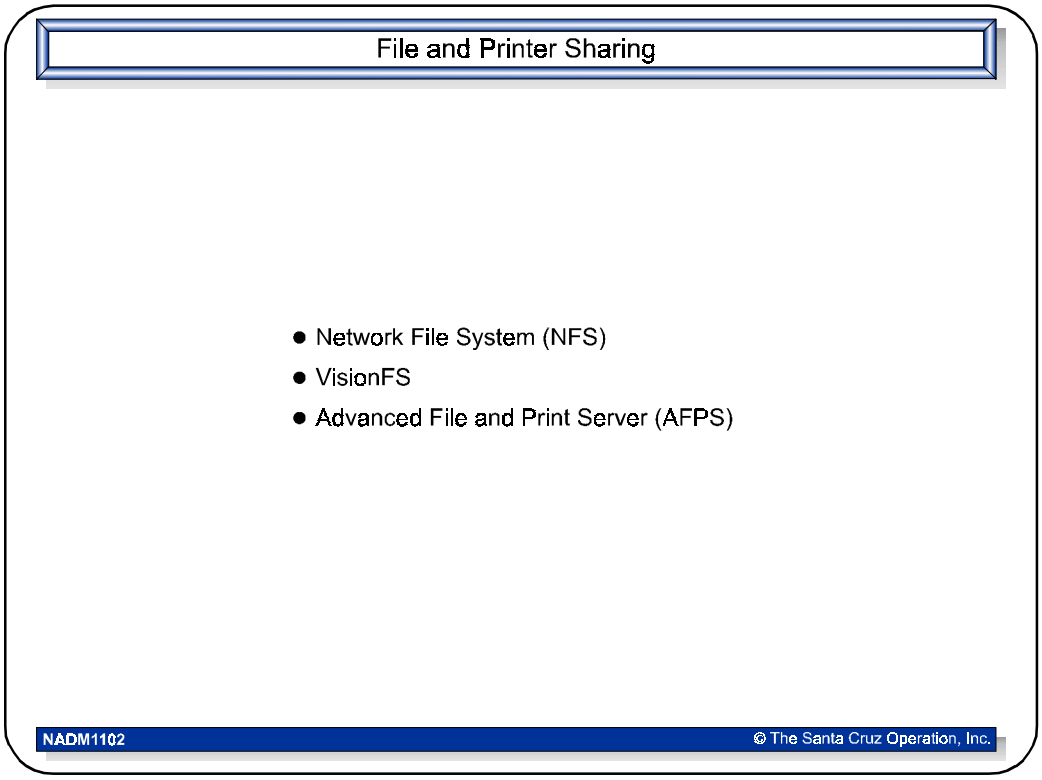
Objectives

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 three 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™ to run UNIX commands from a Windows client
- Use DHCP to provide remote boot configuration services

AV 11-1: Windows Connectivity
Use this slide to preview the lesson contents.

AV 11-2



Instructor Guide

11.1 File and Printer Sharing Overview

Your network can consist of a mixture of servers running UNIX and client systems running Microsoft Windows or another operating system. The UNIX server can store files that you want to make available to your client systems. In addition, you might want users of the client systems to print to printers attached to your UNIX server. OpenServer provides several alternatives for both tasks.

To provide file and printer sharing, you will:

- Select a network protocol for exchanging the relevant data.
- Install client-side software that implements the protocol(s) selected.
- Install server-side software that implements the protocol(s) selected.

In this section, consider three approaches for providing file and printer sharing from an OpenServer system, which include:

- Network File System (NFS®) (file sharing only)
- VisionFS
- SCO Advanced File and Print Server (AFPS)

AV 11-2: File and Printer Sharing
This lists the options for sharing files and printers discussed in this section.

It is not expected that NFS or Advanced File and Print Server will be demonstrated in the classroom. Of course, depending on the time available, student interest, instructor experience, and software availability, you can choose to use any of these for lab exercises.

11.1.1 NFS

SCO OpenServer includes both server and client support for the NFS file sharing protocol. Note that NFS does not provide printer sharing.

The advantage of using NFS to provide file sharing is the wide availability of NFS client and server software for UNIX and non-UNIX operating systems.

The disadvantage of using NFS is that Microsoft operating environments do not provide built-in NFS support.

- NFS client software is not included with Windows 95, Windows for Workgroups, or Windows NT. To use NFS with these operating systems, you 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 requires training all of your users with a new interface.

- NFS is designed for file sharing between UNIX systems. It does not 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. See Appendix B: “NFS” more information.

11.1.2 VisionFS

In most cases, the simplest and most economical option for providing file and printer sharing for Windows clients is VisionFS. VisionFS is a software package for your SCO OpenServer system that 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.

CIFS is a new name for the Server Message Block(SMB) protocol. It is an open, documented protocol, whereas for several years Microsoft made changes to SMB without documenting the changes.

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.

- The VisionFS server software is included with OpenServer at no additional cost. Each user license you purchase for OpenServer includes one client license for VisionFS.

- 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.

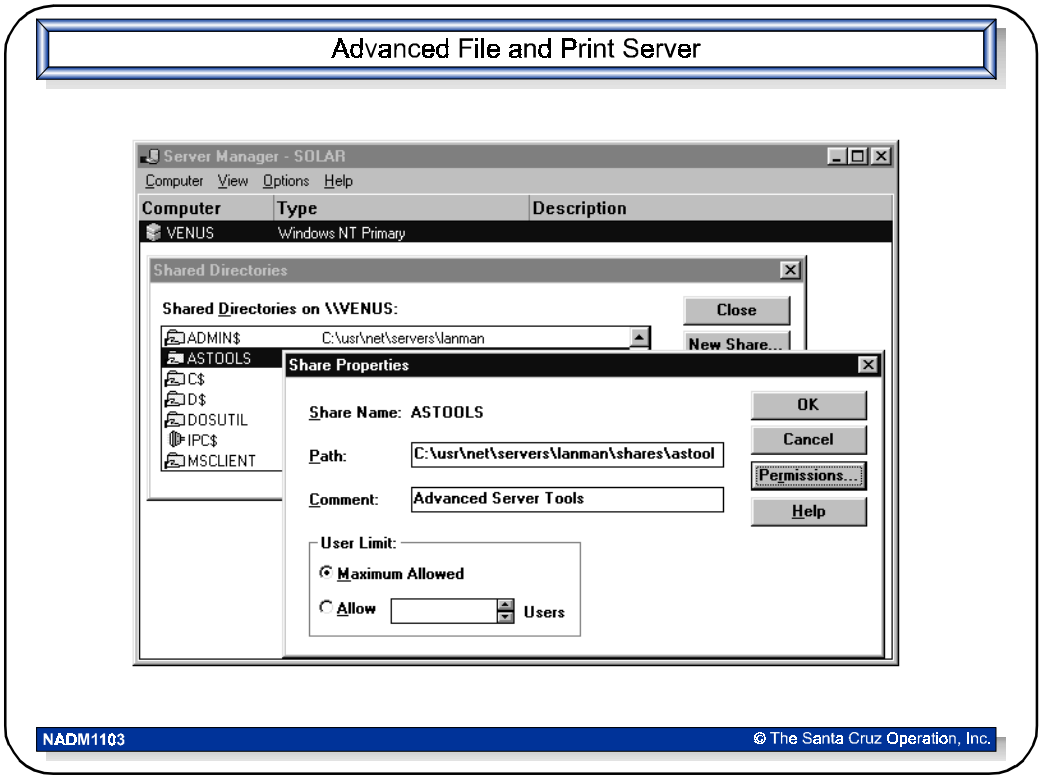
Disadvantages of using VisionFS include:

- 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 OpenServer 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.

AV 11-3


Advanced File and Print Server
Server Manager - SOLAR
Computer View Options Help
Computer Type Description
VENUS Windows NT Primary
Shared Directories
Shared Directories on \\VENUS:
Close
New Share...
ADMIN$ C:\usr\net\servers\lanman
ASTOOLS
C$
D$
DOSUTIL
IPC$
MSCLIENT
Share Properties
Share Name: ASTOOLS
Path: C:\usr\net\servers\lanman\shares\astool
Comment: Advanced Server Tools
OK
Cancel
Permissions...
Help
User Limit:
Maximum Allowed
Allow
Users
NADM1103
© The Santa Cruz Operation, Inc.

11.1.3 Advanced File and Print Server

VisionFS is not the only CIFS server package available for SCO OpenServer Release 5. One 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. A one-user evaluation version of AFPS is included on the OpenServer installation CD-ROM.

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.

VisionFS can authenticate access to shared files or printers against an NT server, but cannot by itself authenticate users as they log in to a Windows 95 system.

- 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.

The disadvantage of AFPS relative to VisionFS is that AFPS can be more complicated to administer if the advance features are used.

AV 11-3: The AFPS Server Manager, running on Windows 95
The Windows tools used to administer AFPS are identical to the native Windows NT tools.

You might also want to mention SAMBA, a freeware product that can also provide SMB/CIFS file and printer sharing from a UNIX server. For more information see the web page at URL *http://samba.anu.edu.au/samba*.

11.2 Installing and Using VisionFS

During 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 Installing VisionFS

To install VisionFS after inserting the Optional Services CD-ROM:

Step 1. Start the **Software Manager.**

Step 2. Select **Software** ⇨ **Install New.**

Step 3. Specify your machine as the host and click on **Continue**.

Step 4. Specify "CD-ROM" as the media device and click on **Continue**.

Step 5. Select "SCO VisionFS" when the list of products is displayed, and click on **Install**.

Step 6. Enter "**y**" to accept the defaults. Press ENTER.

This sets the following:

Authentication	UNIX passwords are used to authenticate access to the server.
Administrator	The VisionFS administrator user name is set to **root**.
Server	The VisionFS server name is set to the UNIX server name.

Step 7. Press ENTER to continue.

Step 8. Enter "**y**" to accept the defaults. Press ENTER.

This will install a 30-day trial license. A full license can be installed later. The license number is provided with the License Kit.

Step 9. Press ENTER to continue. The **Software Manager** copies the VisionFS files to the hard disk.

Step 10. Press ENTER to continue.

After the files are copied, the VisionFS setup screen displays a message indicating that a 30-day license has been installed.

Step 11. Click on **OK**.

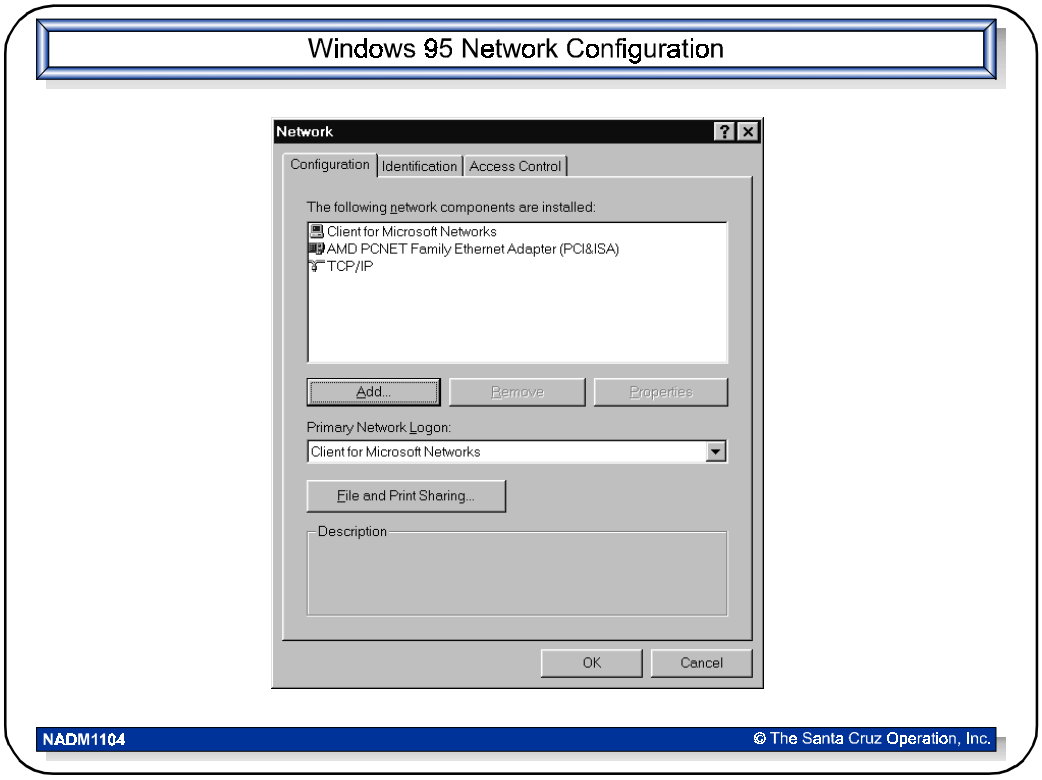
The **Software Manager** completes the installation process by verifying file permissions.

Exercise: Install VisionFS

Run the **Software Manager** to determine if VisionFS is installed on your system.

If it is not present, install VisionFS and accept the default configuration options.

AV 11-4


Windows 95 Network Configuration
Network
Configuration
Identification
Access Control
The following network components are installed:
Client for Microsoft Networks
AMD PCNET Family Ethernet Adapter (PCI&ISA)
TCP/IP
Add...
Remove
Properties
Primary Network Logon:
Client for Microsoft Networks
File and Print Sharing...
Description
OK
Cancel
NADM1104
© The Santa Cruz Operation, Inc.

11.2.2 Windows 95 network configuration

Although all of the software needed to work with VisionFS is included with Windows 95 and other versions of Windows, it might not be installed or correctly configured on every client system.

Follow some or all of the following steps to enable your Windows 95 system to access your VisionFS server.

Note: Only add an adapter, protocol, or client if it is not already installed.

These steps use Windows 95 as an example. You might need to modify them if you are using other versions of Windows or other client operating systems.

There is a problem with Windows 98 and Windows NT Workstation with service pack 3. 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, obtain Technical Article TA 108198 from the SCO Support Web site.

Step 1. Boot Windows 95, and, if needed, log in as any Windows user.

Step 2. Select **Start** ⇨ **Settings** ⇨ **Control Panel**.

Step 3. Double-click on the **Network** icon.

Step 4. Select **Add** ⇨ **Adapter** and choose the option that corresponds to the network card installed in your computer. Click on **OK.**

Windows 95 should automatically add NetBEUI, IPX/SPX Compatible Protocol, and Client for NetWare Networks at this point. VisionFS does not use any of these. They can be removed unless they are needed to communicate with another system on the network.

Step 5. Select **Add** ⇨ **Protocol** ⇨ **Microsoft** ⇨ **TCP/IP**.

Step 6. Select **TCP/IP** ⇨ **Properties** and enter the local IP address.

Step 7. Select **Add** ⇨ **Client** ⇨ **Microsoft** ⇨ **Client for Microsoft Networks**, if it is not already listed as being installed.

Step 8. Click on **OK**. Reboot if prompted.

AV 11-4: Windows 95 Network Configuration
This shows a network adapter, the TCP/IP protocol, and the CIFS client all installed.

To access your VisionFS server, double-click on the **Network Neighborhood** icon.

If your VisionFS server is listed, double-click on its icon.

If your VisionFS server is not listed:

Step 1. Select **Start** ⇨ **Run** from the Windows 95 menu.

Step 2. In the "Open" field, enter:

`\\servername`

where *servername* is the name of your VisionFS server.

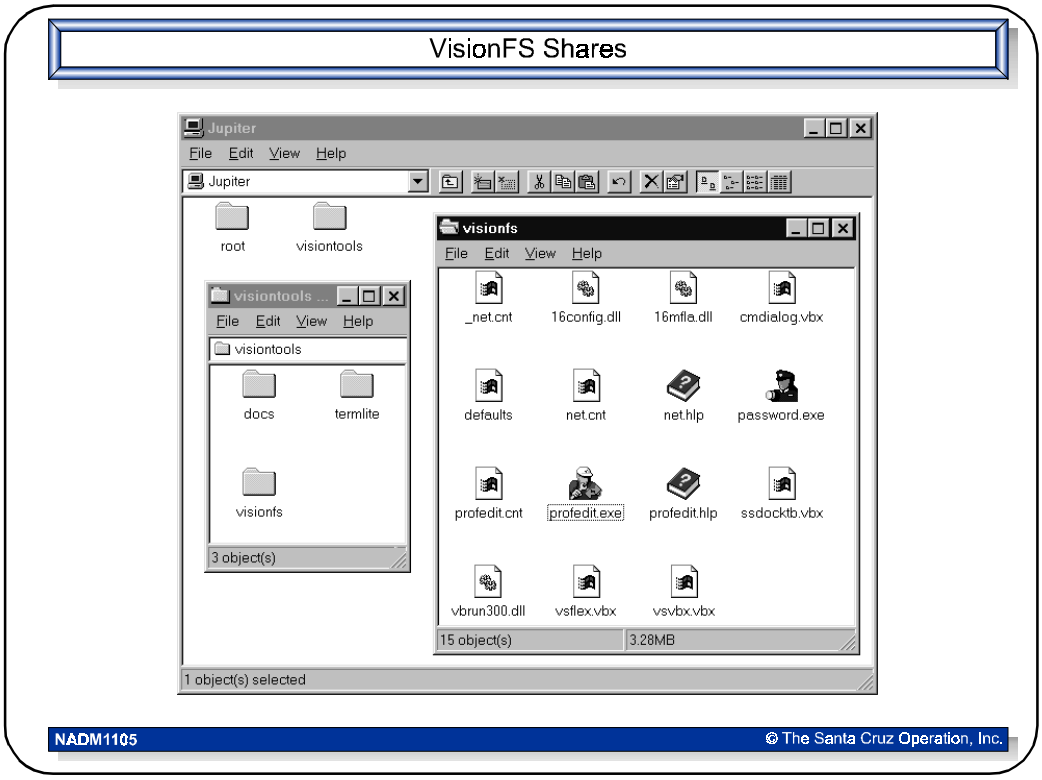
If your network uses workgroups, you might want to configure your Windows 95 system to be in the workgroup normally used in the classroom. Select **Start** ⇨ **Settings** ⇨ **Control Panel** ⇨ **Network** ⇨ **Identification** and enter the workgroup name in the appropriate field. This might cause your VisionFS server to appear in the **Network Neighborhood** window when it would not otherwise.

Exercise: Configure Windows 95 for VisionFS

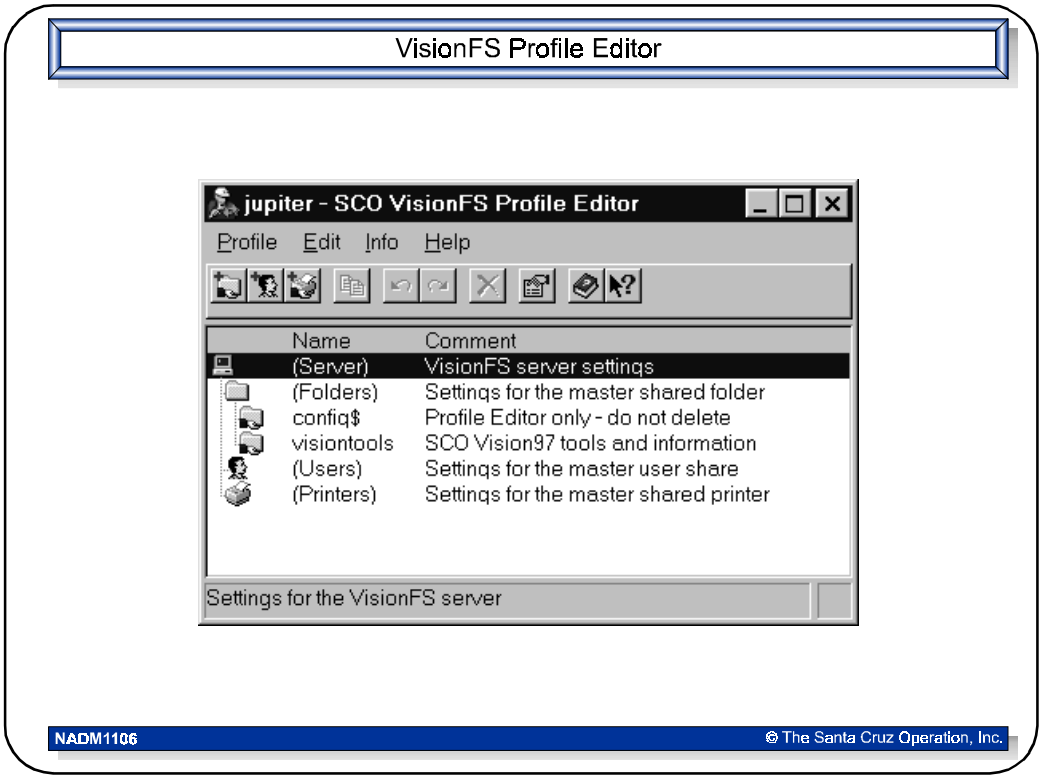
Configure your Windows 95 system so that it can access your VisionFS server. Obtain information such as network interface card model, hardware configuration, and an IP address from your instructor.

You should see the *visiontools* share (possibly among others) on the VisionFS server. You might not be able to read or write to any shared directories because of permissions.

AV 11-5


VisionFS Shares
Jupiter
File Edit View Help
Jupiter
root
visiontools
visionfs
File Edit View Help
_net.cnt
16config.dll
16mfla.dll
cmdialog.vbx
defaults
net.cnt
net.hlp
password.exe
profedit.cnt
profedit.exe
profedit.hlp
ssdocktb.vbx
vbrun300.dll
vsflex.vbx
vsvbx.vbx
15 object(s)
3.28MB
visiontools ...
File Edit View Help
visiontools
docs
termlite
visionfs
3 object(s)
1 object(s) selected
NADM1105
© The Santa Cruz Operation, Inc.

AV 11-6


VisionFS Profile Editor
jupiter - SCO VisionFS Profile Editor
Profile Edit Info Help
Name
Comment
(Server)
VisionFS server settings
(Folders)
Settings for the master shared folder
config$
Profile Editor only - do not delete
visiontools
SCO Vision97 tools and information
(Users)
Settings for the master user share
(Printers)
Settings for the master shared printer
Settings for the VisionFS server
NADM1106
© The Santa Cruz Operation, Inc.

11.2.3 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 will prompt you for the UNIX **root** password when you try to access shared directories or printers.

If your Windows client system uses a Windows NT domain controller to authenticate users, then you will need to add a **root** user to the domain controller, keeping the password the same as the UNIX root password.

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.

The **Profile Editor** is shared as *visiontools\visionfs\profedit.exe.*

AV 11-5: VisionFS Shares
This shows the default shares on a freshly installed VisionFS server.

AV 11-6: The VisionFS Profile Editor
This is the default configuration for a freshly installed server.

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 the following 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 **root**.

Step 2. Access the VisionFS server.

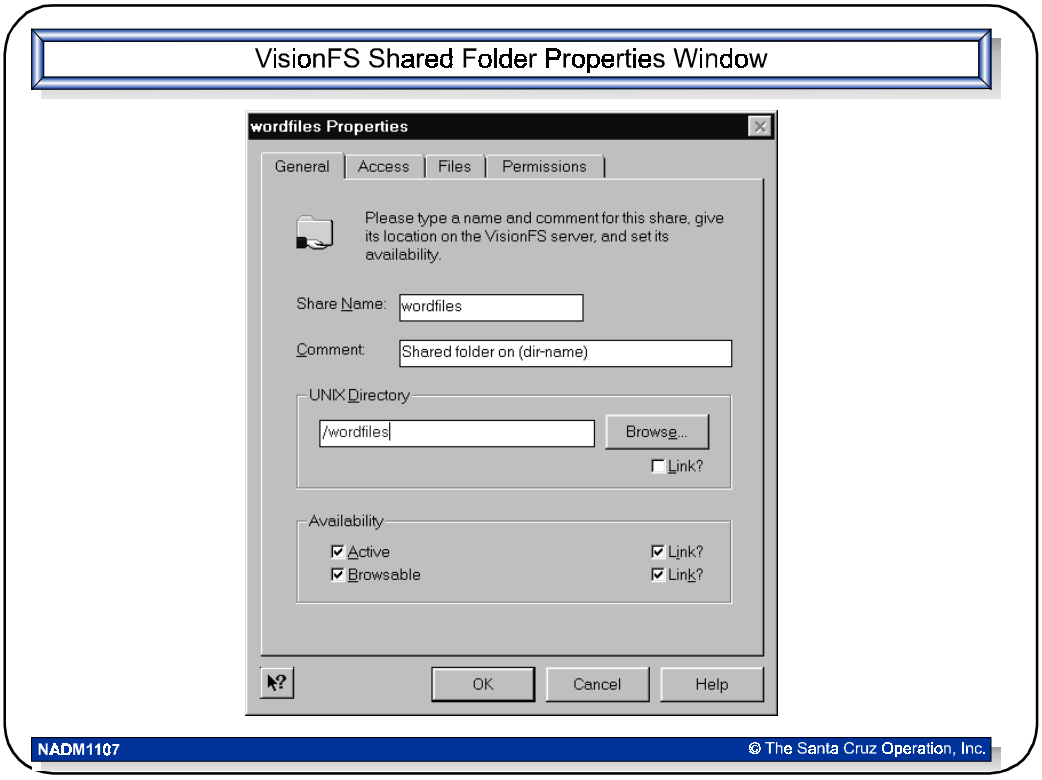
Step 3. Open the *visiontools* share.

Step 4. Open the *visionfs* folder.

Step 5. Run the VisionFS **Profile Editor**, execute:

profedit.exe

AV 11-7


VisionFS Shared Folder Properties Window
wordfiles Properties
General
Access
Files
Permissions
Please type a name and comment for this share, give its location on the VisionFS server, and set its availability.
Share Name:
wordfiles
Comment:
Shared folder on (dir-name)
UNIX Directory
/wordfiles
Browse...
Link?
Availability
Active
Browsable
Link?
Link?
OK
Cancel
Help
NADM1107
© The Santa Cruz Operation, Inc.

11.2.4 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

Depending on your locale and language configuration, the last component of the *visiontools* directory on the server can be different from *en_US*.

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.

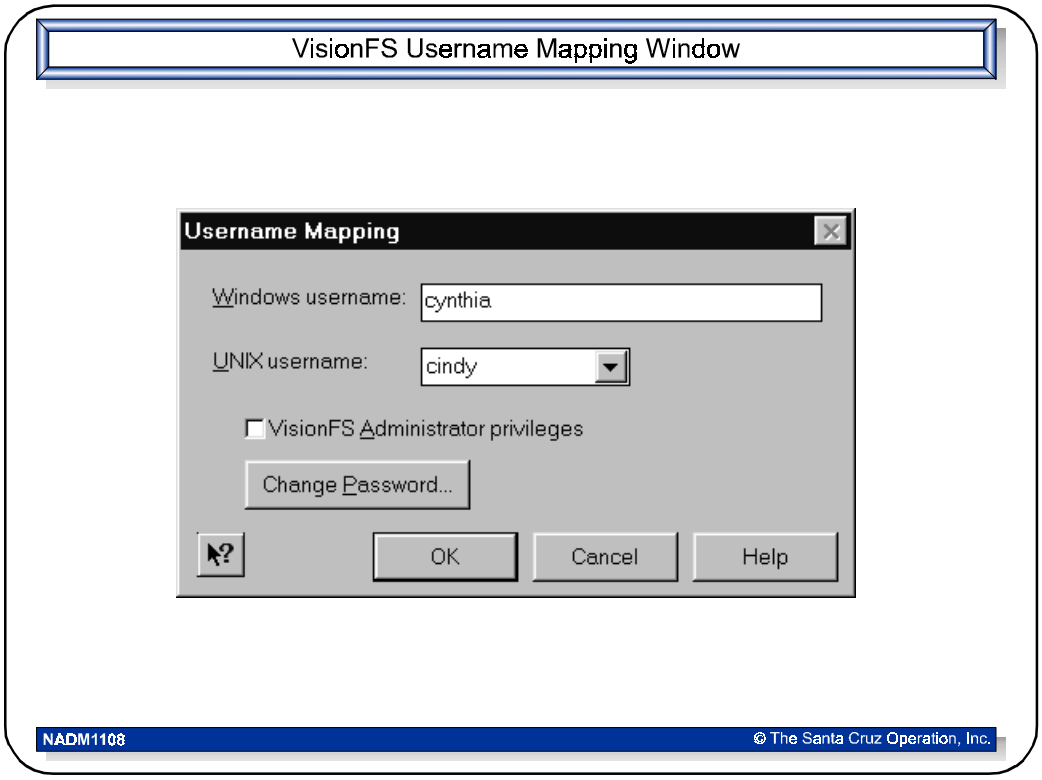
Do not have two people updating the same server at the same time, as this will cause problems.

AV 11-7: VisionFS Shared Folder Properties Window
The slide shows the fields filled in as they should be for the following exercise.

Exercise: Sharing a directory

On your OpenServer system, create a directory called */wordfiles*. Use the **Profile Editor** to share this directory with the same share name.

AV 11-8


VisionFS Username Mapping Window
Username Mapping
Windows username:
cynthia
UNIX username:
cindy
VisionFS Administrator privileges
Change Password...
OK
Cancel
Help
NADM1108
© The Santa Cruz Operation, Inc.

11.2.5 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**.

- If the **Link** button is checked, the setting is automatically set to the default value stored in the master shared folder.
- 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.

11.2.5.1 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.

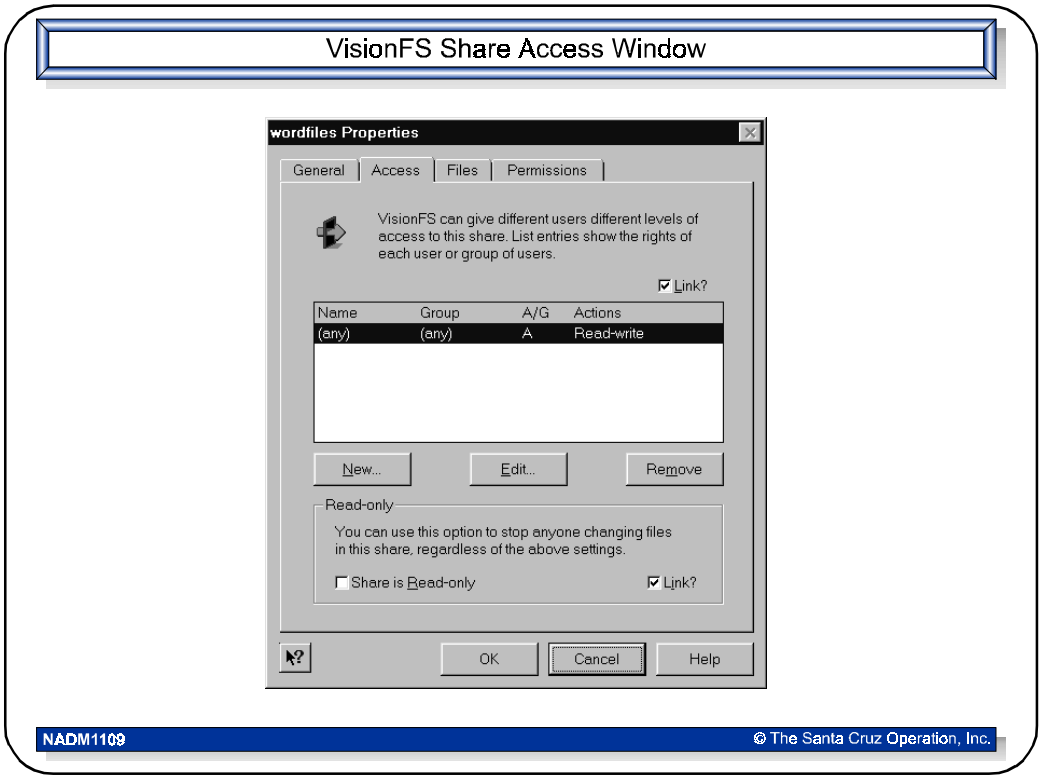
Step 6. Enter **cindy** in the “UNIX username” field.

Step 7. Click on **OK**.

AV 11-8: The VisionFS Username Mapping Window
This slide shows the **Users** tab in the VisionFS **Server Properties** window, which is used to map Windows user names to UNIX user names.

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 11-9


VisionFS Share Access Window
wordfiles Properties
General
Access
Files
Permissions
VisionFS can give different users different levels of access to this share. List entries show the rights of each user or group of users.
Link?
Name
Group
A/G
Actions
(any)
(any)
A
Read-write
New...
Edit...
Remove
Read-only
You can use this option to stop anyone changing files in this share, regardless of the above settings.
Share is Read-only
Link?
OK
Cancel
Help
NADM1109
© The Santa Cruz Operation, Inc.

11.2.5.2 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 how a user is allowed to access shares, you must consider:

- VisionFS share access is configured with the VisionFS **Profile Editor**.
- Normal UNIX permissions are configured with the **chmod**(C), **chown**(C), and **chgrp**(C) 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 in AV 11-9.

AV 11-9: VisionFS Share Access Window
This slide shows the default access permissions for a new share.

AV 11-10



AV 11-11



For example, to change the default access to read-only for a share:

Step 1. Select the line corresponding to the “Name” or **(any)**.

Step 2. Click on **Edit**.

Step 3. Select **Read only** under the “Actions” heading.

Step 4. Click on **OK**.

Note that you can also select “Share is read-only” on the Access panel for the share. However, that option applies to all users and cannot be overridden for a particular user. The method described here can be overridden.

To add read-write access for a particular user named **ben**, as shown in AV 11-10:

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.

AV 11-10: VisionFS User Access Rights
This slide shows the User Access Rights window filled out to grant **ben** read-write access.

To grant read-only access to guest users as shown in AV 11-11:

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 are used for the guest users under the “Performed as UNIX user” heading.

Step 5. Click on **OK**.

AV 11-11: VisionFS User Access Rights
This slide shows the User Access Rights window filled out to grant read-only access to guest users, with all guest actions performed as the UNIX user **cindy**.

11.2.6 Sharing printers with 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.

11.2.6.1 Sharing server printers with 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.

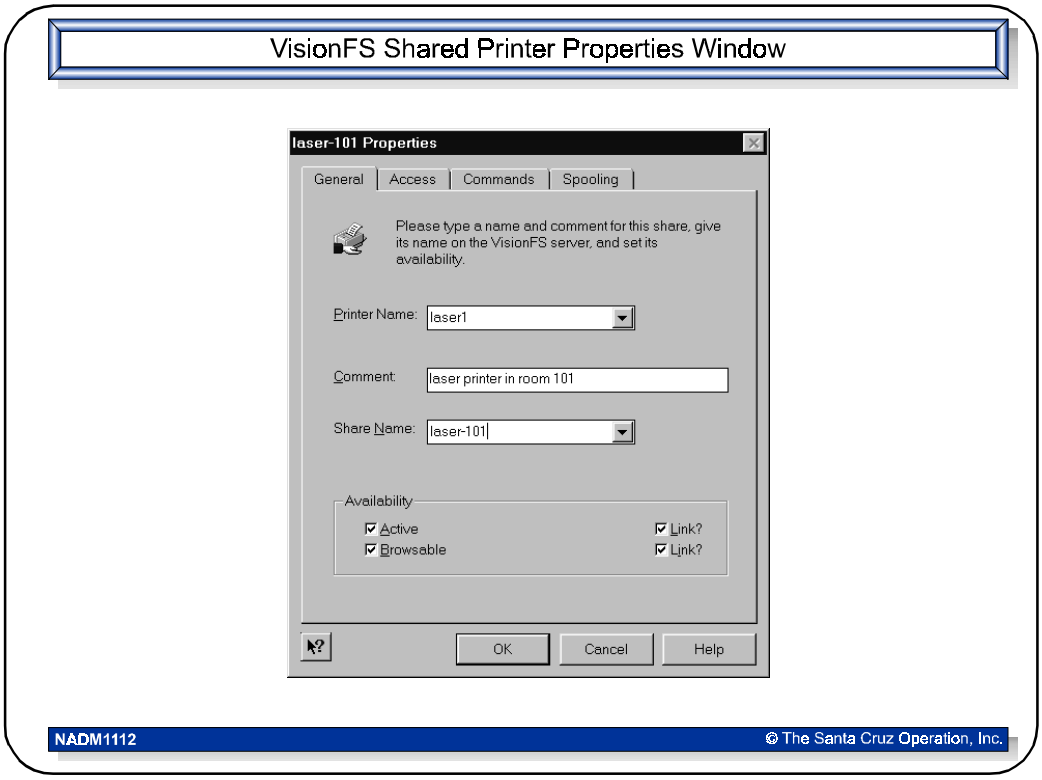
Note that the VisionFS server normally starts when the UNIX system boots. If you have added new printers since the last reboot, restart VisionFS so that it detects the new printers.

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
```

AV 11-12


VisionFS Shared Printer Properties Window
laser-101 Properties
General
Access
Commands
Spooling
Please type a name and comment for this share, give its name on the VisionFS server, and set its availability.
Printer Name:
laser1
Comment:
laser printer in room 101
Share Name:
laser-101
Availability
Active
Browsable
Link?
Link?
OK
Cancel
Help
NADM1112
© The Santa Cruz Operation, Inc.

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.**

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**.

Customize settings for a particular printer. First create an explicit share, rather than use 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**.

AV 11-12: VisionFS Shared Printer Properties Window
This slide shows the UNIX printer **laser1** shared with the name **laser-101**.

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.

11.2.6.2 Sharing client printers with 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 your printers.
2. Configure Windows to share a particular printer.
3. Use VisionFS to access the shared client printer.

For these steps, Windows 95 is used for the client operating system. Other versions of Windows and other operating systems can have different steps.

To share your 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

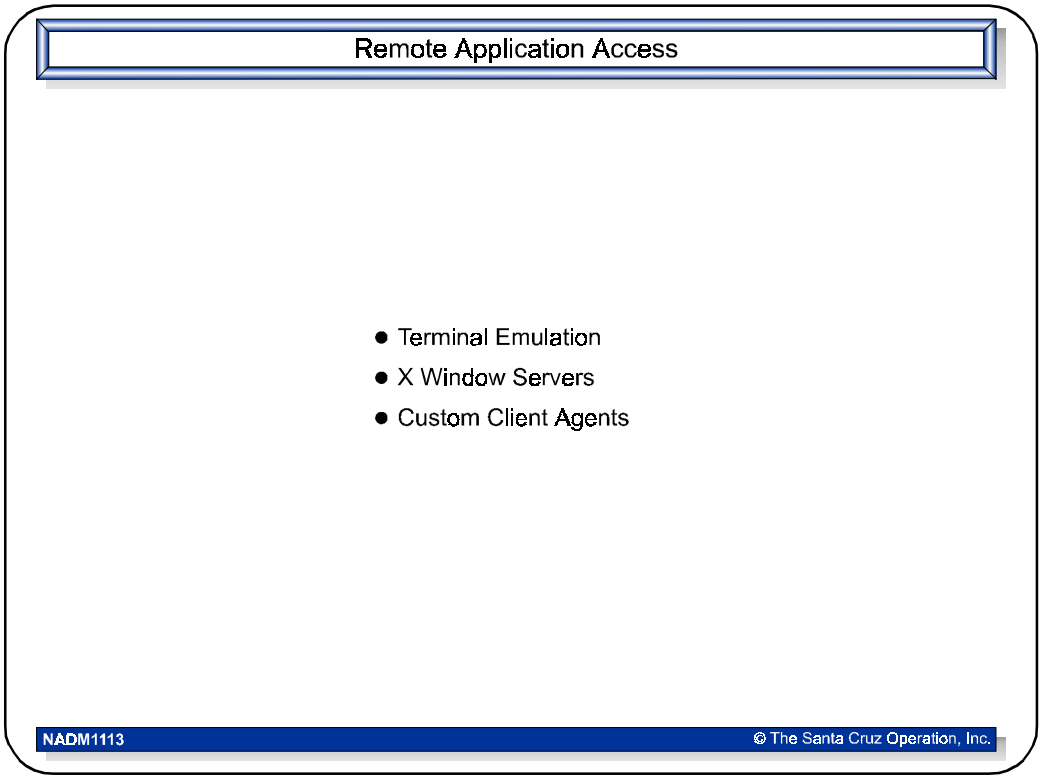
printer Is the share name for the printer

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 choose to write a shell script that runs the above command. Alternatively, you could write a printer interface script so the shared printer could be accessed using the **lp**(C) command.

AV 11-13


Remote Application Access
Terminal Emulation
X Window Servers
Custom Client Agents
NADM1113
© The Santa Cruz Operation, Inc.

11.3 Remote Application Access

In addition to (or instead of) sharing files and printers, you can also 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 need to provide access to an application to many people at once.

There are a variety of strategies for providing client users with access to UNIX applications. The alternatives discussed in the section are:

- Terminal Emulation
- X Window Servers
- Custom Client Agents

AV 11-13: Remote Application Access
This slide lists the alternatives for providing remote application access discussed in this section.

Of the following topics, only TermLite is expected to be available in the classroom and used hands-on.

Depending on available time, student interest, instructor experience, and software availability, you can add extra exercises or labs in this section. Other programs to use include SCO TermVision™ and XVision®. Third-party terminal emulators, X Window® servers, and custom client/server applications can also be good candidates for lab use.

11.3.1 Terminal emulation

If your UNIX 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.

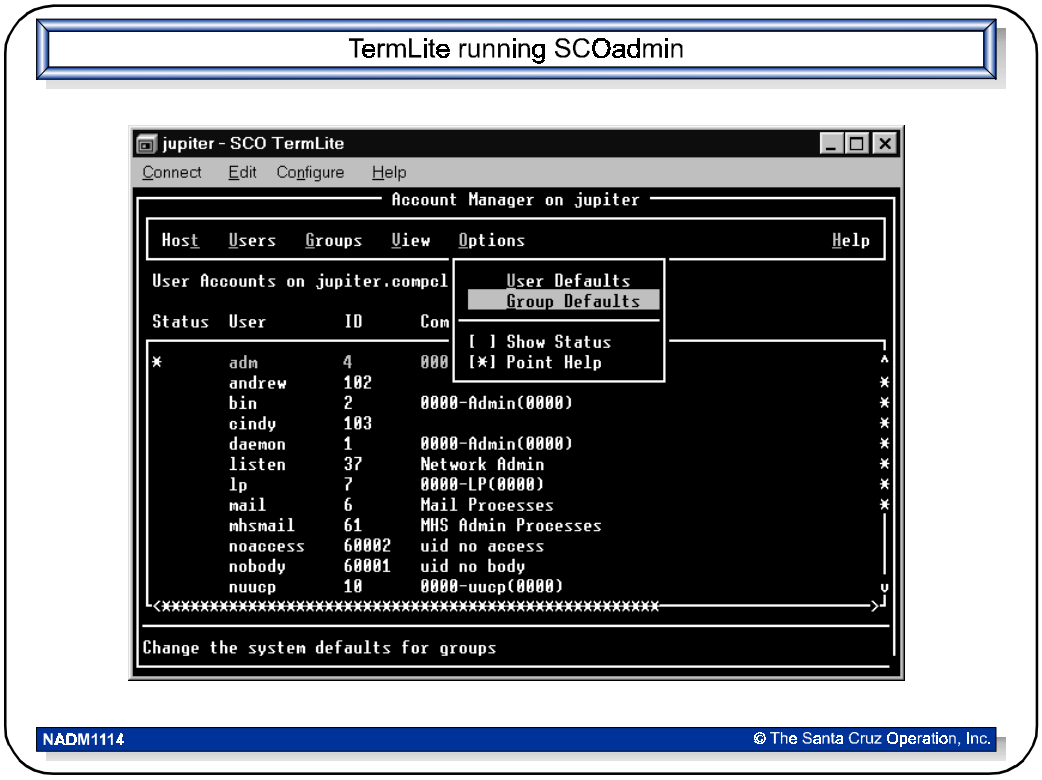
Advantages of terminal emulation include:

- 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 greatly simplifies installation and support over a custom client approach.
- Your users can use a very 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 running SCOadmin
jupiter - SCO TermLite
Connect Edit Configure Help
Account Manager on jupiter
Host Users Groups View Options Help
User Accounts on jupiter.compcl
Status User ID Com
User Defaults
Group Defaults
[ ] Show Status
[*] Point Help
* adm 4 000
andrew 102
bin 2 0000-Admin(0000)
cindy 103
daemon 1 0000-Admin(0000)
listen 37 Network Admin
lp 7 0000-LP(0000)
mail 6 Mail Processes
mhsmail 61 MHS Admin Processes
noaccess 60002 uid no access
nobody 60001 uid no body
nuucp 10 0000-uucp(0000)
Change the system defaults for groups
NADM1114
© The Santa Cruz Operation, Inc.

TermLite

TermLite is a lightweight terminal emulator for Windows that is included with OpenServer Release 5. It emulates the ansi, VT100, and VT320 terminal types, among others.

A version of TermLite is installed on your OpenServer system when VisionFS is installed, and made available in the VisionFS shared folder. To run TermLite from Windows 95, you have two options:

- Run TermLite directly from the share, leaving the executable file on the VisionFS server. Select **Network Neighborhood ⇨ server ⇨ visiontools ⇨ termlite ⇨ termlite.exe**

  This can also be done using the MS-DOS command line (or using **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. To do this, execute:

  `\\server\visiontools\termlite\setup.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.

AV 11-14: TermLite running SCOadmin
This slide shows SCOadmin running in a TermLite window, using the VT100 terminal type.

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 such 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**(C), **scoadmin**(ADM), and **setcolor**(C). Adjust the terminal type if needed.

SCOadmin might not work correctly unless VT100 terminal emulation is selected.

Step 3. Select **Configure** ⇨ **Terminal** from the menu.

- Choose yellow text on a blue background.
- Change the number of lines displayed to 35.

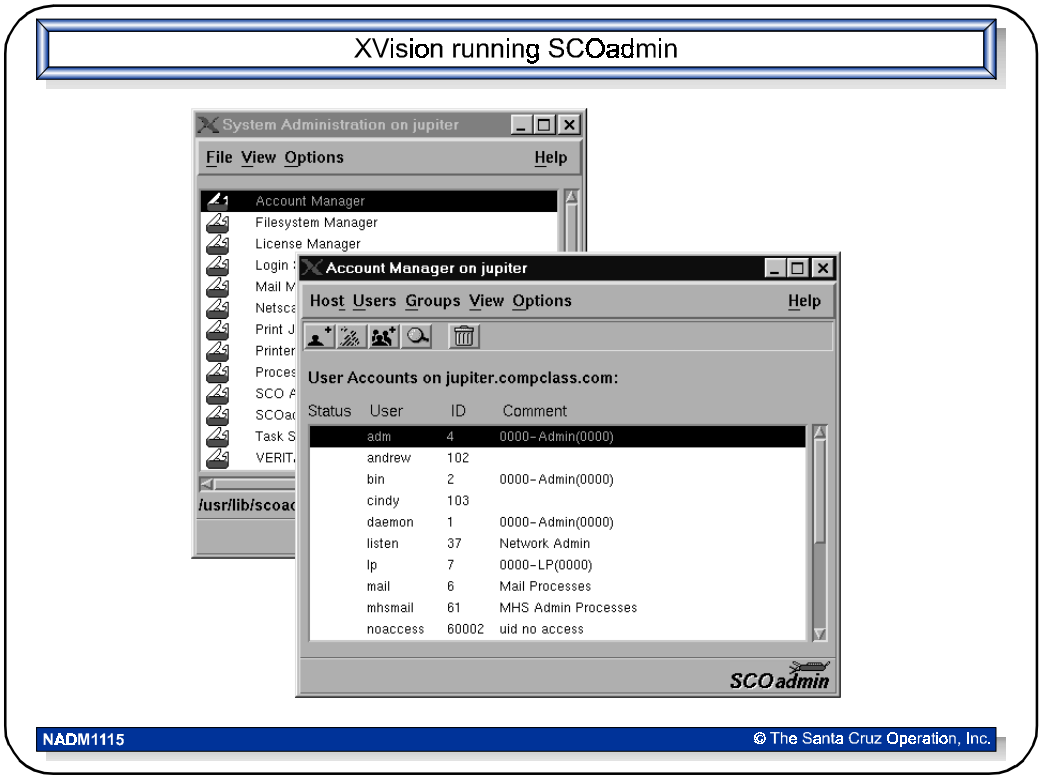
Step 4. Resize the window so the font is a readable size.

Step 5. Experiment with other TermLite configuration options.

Instructor Guide

You cannot run TermLite directly from the OpenServer Optional Services CD-ROM; you must install the Termlite package on your OpenServer system first. TermLite is branded/licensed as it copies to the Windows client. If you want to use TermLite without VisionFS, use **ftp** to transfer */usr/vision/bin/pctools/en_US/termlite/Termlite.exe* to a Windows system. It will run in evaluation mode and expire after 30 days.

AV 11-15


XVision running SCOadmin
System Administration on jupiter
File View Options
Help
Account Manager
Filesystem Manager
License Manager
Account Manager on jupiter
Host Users Groups View Options
Help
User Accounts on jupiter.compclass.com:
Status User ID Comment
adm 4 0000-Admin(0000)
andrew 102
bin 2 0000-Admin(0000)
cindy 103
daemon 1 0000-Admin(0000)
listen 37 Network Admin
lp 7 0000-LP(0000)
mail 6 Mail Processes
mhsmail 61 MHS Admin Processes
noaccess 60002 uid no access
SCOadmin
NADM1115
© The Santa Cruz Operation, Inc.

11.3.2 X Window servers

AV 11-15: XVision running SCOadmin
This slide shows XVision being used to run SCOadmin on a Windows 95 system.

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 was designed to allow the client application and the display server to run on different machines connected by a network. It consists of a client/server pair:

- The *client* is the application that requires access to a graphic display.
- The *server* is software that draws on a graphic display. The X server software must run on the machine that provides the graphic display.

SCO OpenServer includes both X Windows server and client software. This allows one OpenServer system to run an X client application and another OpenServer system to display the graphical interface for the client.

To use Microsoft Windows as the display platform, you need an X Windows server that accepts network requests to display graphics.

For example, you might want to administer your OpenServer system from a Windows workstation using the **scoadmin**(ADM) graphical interface. The OpenServer system would run the SCOadmin client application (the **Printer Manager**, for example) and display on the Windows system.

One example of X Window server software for Microsoft Windows is SCO XVision. (It is not included with OpenServer.)

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. It 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. 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:

- A separate client agent is required for every application, which increases the complexity of installation and support.
- 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.

Java is a programming language used to write custom mini-applications called Java applets. The applets can be integrated into Web pages and downloaded from a server. Any client that has a web browser (as many do) requires no additional software installation to run the applet.

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. SCO OpenServer can help by providing remote boot configuration services to your Windows (and other operating system) clients.

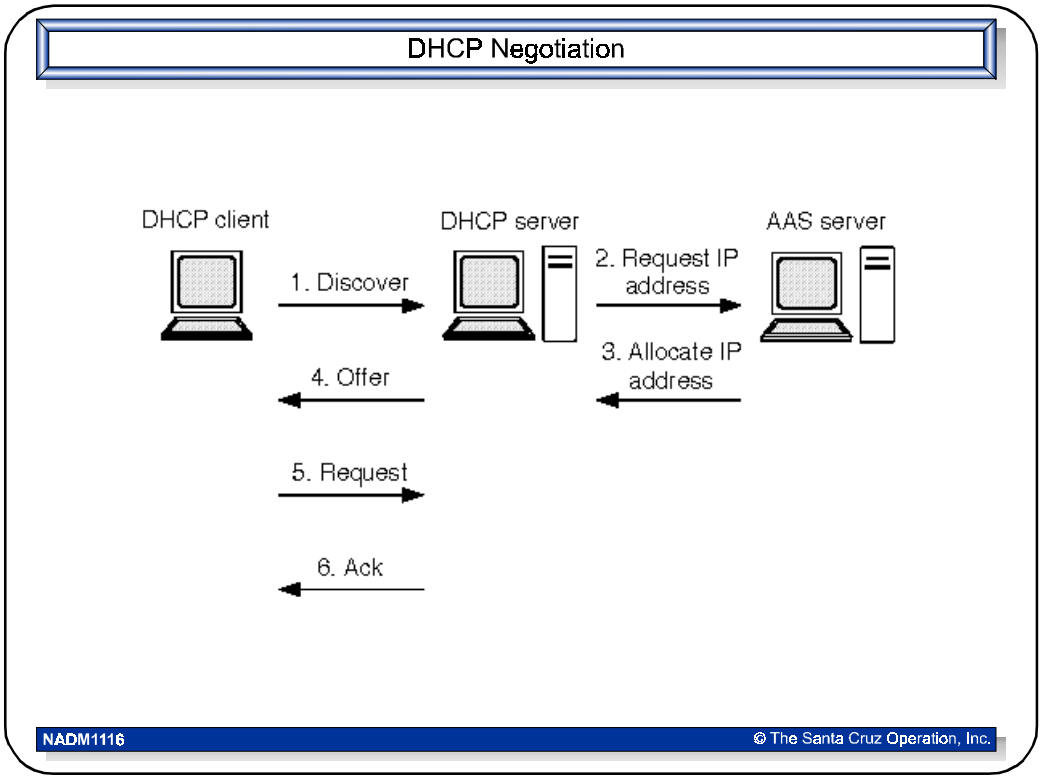
This section describes the use of the Dynamic Host Configuration Protocol (DHCP). A DHCP server answers broadcasts from a booting client with configuration parameters for the client to use.

Examples of the parameters DHCP can provide include IP address, netmask, broadcast address, domain name, DNS server, and default router.

The 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.

AV 11-16


DHCP Negotiation
DHCP client
DHCP server
AAS server
1. Discover
2. Request IP address
3. Allocate IP address
4. Offer
5. Request
6. Ack
NADM1116
© The Santa Cruz Operation, Inc.

AV 11-16: DHCP Negotiation
This slide shows the steps in serving configuration parameters to a DHCP client.

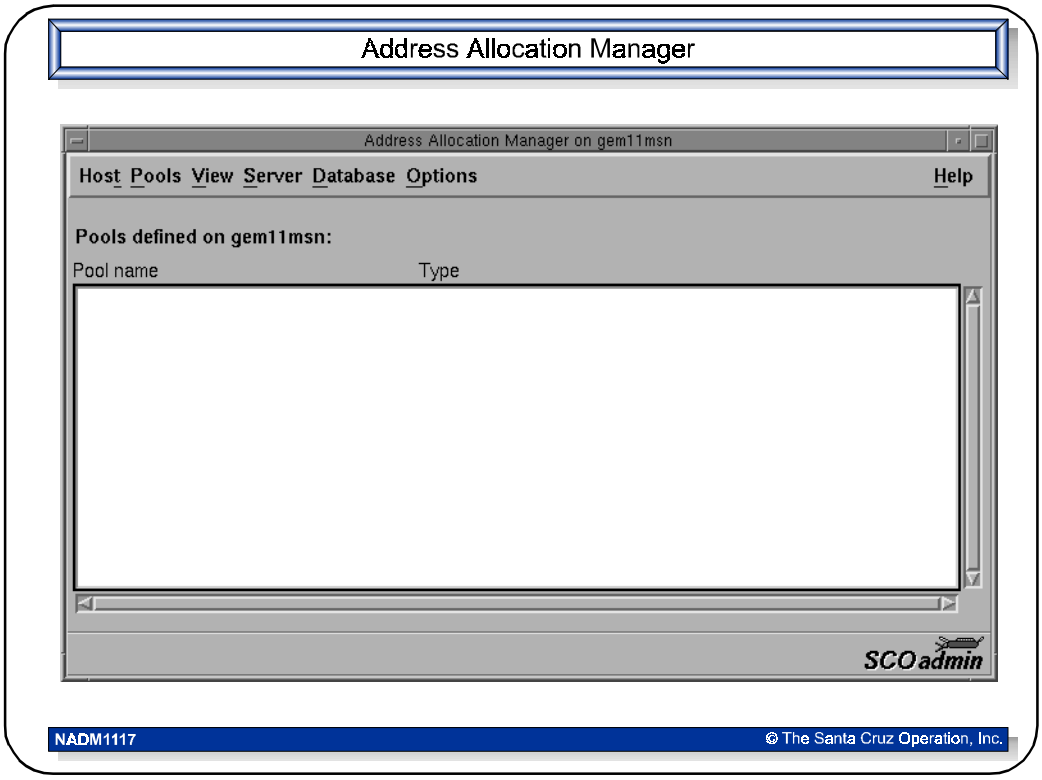
When a DHCP client boots, the following sequence occurs:

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.

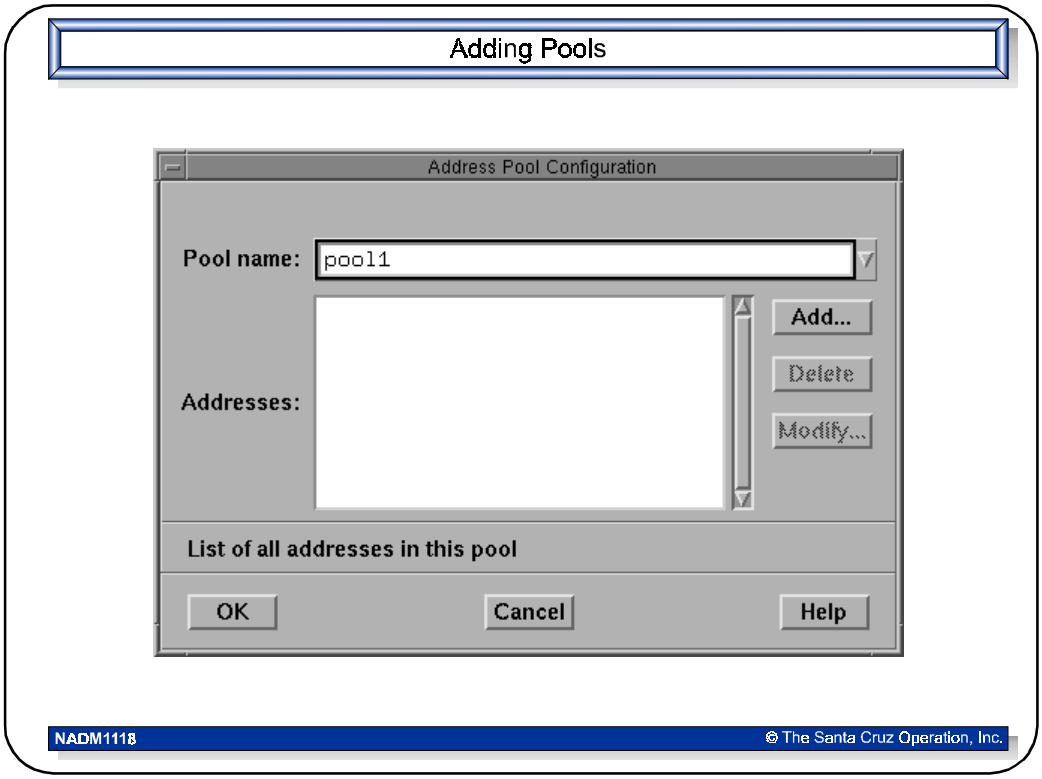
The basic procedure for configuring a server to provide DHCP configuration parameters to a client is:

1. Configure the Address Allocation Server using the **Address Allocation Manager**:
    - Create address pools—define the pools from which DHCP can pick IP addresses that will be assigned to clients.
    - Start the Address Allocation daemon, **aasd.**
2. Configure DHCP using the **DHCP Manager**:
    - Set address lease parameters—the DHCP server *leases* an IP address to a client. The lease might be for a limited or unlimited period of time.
    - Set the options that determine the parameters that can be served to the client. For example, configure the IP addresses of the DNS name servers. Options can be set:
        - Globally—applying to all clients
        - For all clients on a specified subnet
        - Individually
3. Configure the DHCP client to obtain networking parameters from the server.

AV 11-17



AV 11-18



Instructor Guide

11.4.1 Configuring an Address Allocation Server

AV 11-17: Address Allocation Manager
This slide shows the **Address Allocation Manager** window with no pools configured.

The Address Allocation Server can be used to assign addresses for DHCP. To create an address pool:

Step 1. Run the **Address Allocation Manager**. This displays the window shown in AV 11-17.

Step 2. Select **View** ⇨ **Pool Definition**.

Step 3. Select **Pools** ⇨ **Add** to display the window shown in AV 11-17.

Step 4. Supply a name for the pool in the “Pool name” field.

Step 5. Click on **Add** to add addresses to the new pool, specifying a single address or range, as appropriate for your configuration. When finished, click on **OK**.

Step 6. Start the Select **Server** ⇨ **Start Server**.

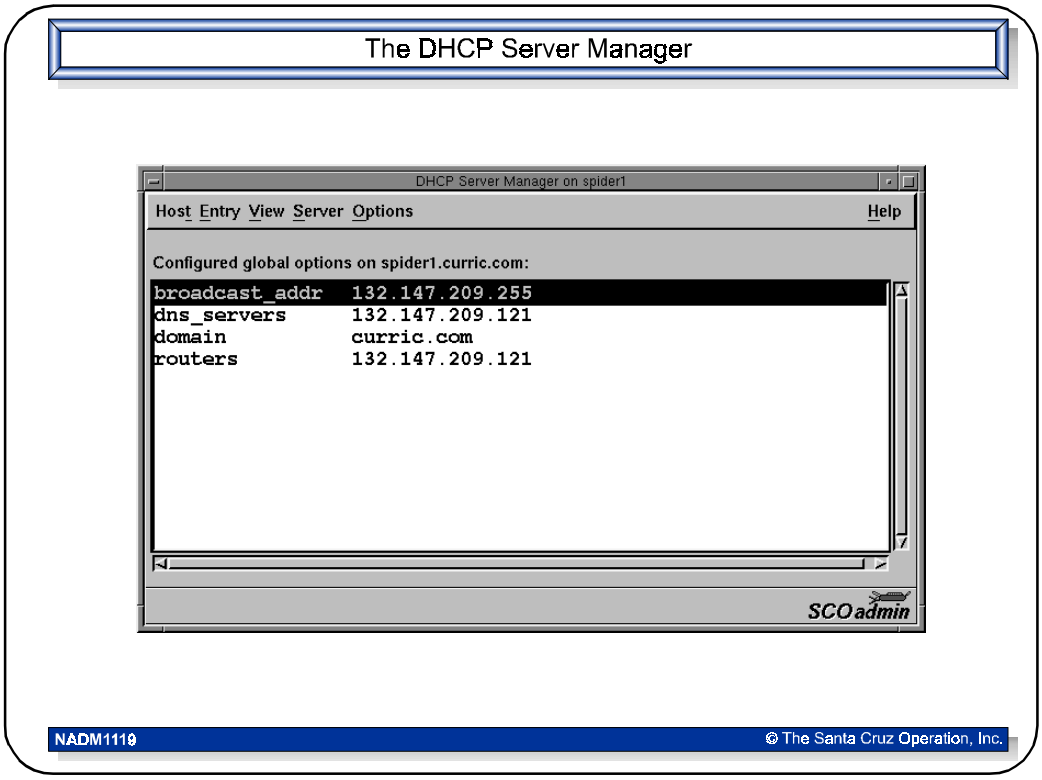
AV 11-18: Address Allocation Manager
This slide shows the **Address Allocation Manager Add Pool** window.

Exercise: Configuring the Address Allocation Server

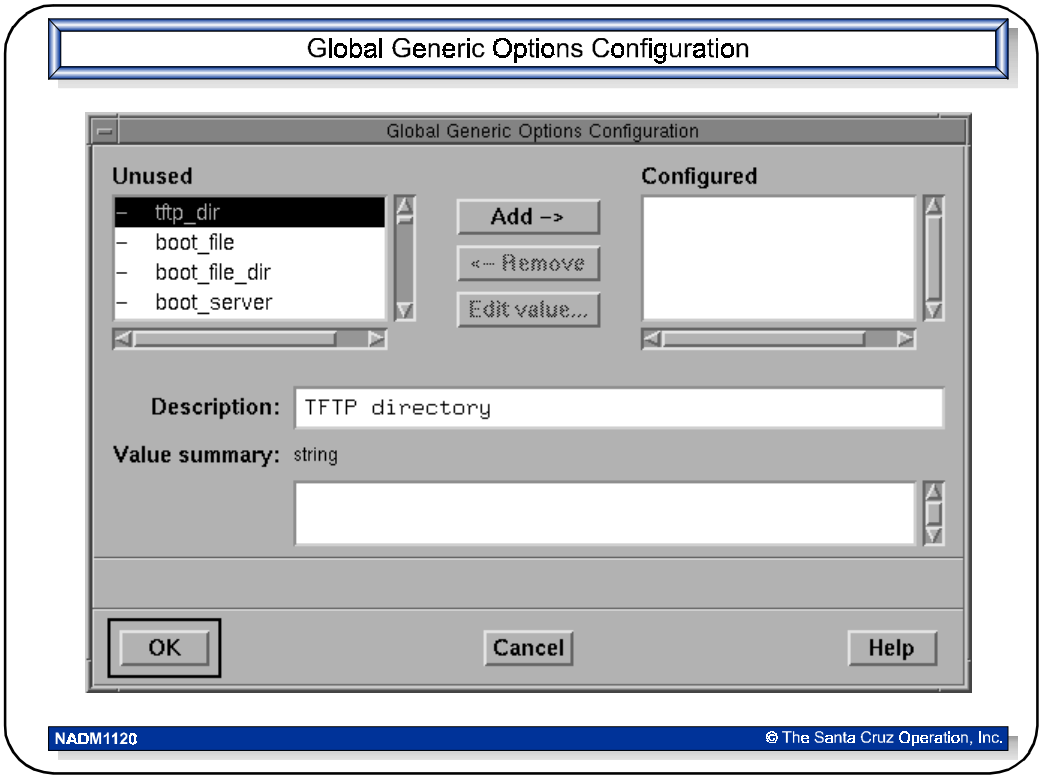
Use the **Address Allocation Manager** to create a new address pool, using instructor-supplied IP addresses.

Start the AAS daemon.

AV 11-19



AV 11-20



11.4.2 Configuring the DHCP server

AV 11-19: DHCP Server Manager
This slide shows the selection **View ⇨ Global options** in the **DHCP Server Manager.**

In addition to configuring IP address lease characteristics, the **DHCP Server Manager** is used to configure the parameters that are supplied to requesting DHCP clients. These global options apply to Windows 95 clients:

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

If you have other types of DHCP clients, see RFC 1533 for a full list of the many global options and their meanings.

AV 11-20: Global Generic Options Configuration
This slide shows the window used to configure DHCP global options.Note that SCO OpenServer cannot currently act as a DHCP client.

To set a value for a DHCP global option:

Step 1. Run the **DHCP Server Manager** and select **View ⇨ Global Options**.

Step 2. Select **Entry ⇨ Add**. This displays the window shown in AV 11-20.

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 and click on **OK**.

Step 6. Configure additional options, if desired.

Step 7. Click on **OK** in the **Global Generic Options Configuration** window.

A bug in OpenServer can result in the message, “Unable to create global definition parameters.” Although your options are saved, the **inetd** daemon may no longer be running. To work around this problem:

- Click on **OK** in the Error window.
- Restart **inetd** at the shell prompt. Enter:
  `/usr/bin/sd inetd`
- Click on **OK** in the Global Generic Options Configuration window.

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 11-21: Global Generic Options Configuration
This slide shows the window used to configure DHCP global options.

11.4.2.1 Configuring Windows 95 as a DHCP client

To configure Windows 95 as a DHCP client system:

Step 1. Log in to a Windows 95 system.

Step 2. Right-click on the **Network Neighborhood** icon and select **Properties** from the pop-up menu.

Step 3. Select **TCP/IP** and click on **Properties**.

Step 4. Select “Obtain an IP address automatically.”

Step 5. Click on the **DNS Configuration** tab and select “Disable DNS.”

Step 6. Click on the **Gateway** tab and remove any installed gateways.

Step 7. Click on **OK** to exit each window.

Step 8. 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 DHCP client

Configure a Windows 95 system to use DHCP as described above.

Run **winipcfg** to examine your assigned network configuration.

11.4.2.2 DHCP and DNS

Currently, there is no interface between DNS and DHCP to dynamically update DNS records because IP addresses are dynamically assigned by DHCP.

The version of BIND in SCO Openserver (8.1.1) does support dynamic updates using the **nsupdate** command (*/etc/nsupdate*). However, DHCP does not make use of this feature.

This can cause problems if:

- A client 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, NFS and trusted access can be affected.

One workaround is to avoid using 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 a default gateway, DNS server, or domain name.

The **nsupdate** command has no manual page. For more information, see the book:

Albitz, Paul & Cricket Liu, ***DNS and BIND***, 2nd Edition, Sebastopol, CA: O'Reilly & Associates, 1997, ISBN: 1-56592-236-0

11.5 Lesson Summary

Today, when it is rare to find a network of computers that all use the same operating system, 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 purchase of additional software beyond SCO OpenServer and Windows 95?

2. What are two alternative ways of 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**(C) on a UNIX system?

4. Describe at least two strategies for providing word processors to Windows users. Consider both file and printer sharing and remote application access strategies. What are the advantages and disadvantages of your strategies?

5. What does the Address Allocation Server do?

6. 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

/usr/vision/bin/visionfs	VisionFS server administration tool
winipcfg	A Windows 95 command for displaying IP configuration settings

Relevant UNIX directories

/usr/vision/bin/pctools/en_US	Home of the **visiontools** shared folder

Relevant VisionFS shared files

visiontools\visionfs\profedit.exe	The VisionFS Profile Editor
visiontools\docs\vfsintro.pdf	***Introducing VisionFS***, documentation in Acrobat format (120+ pages)
visiontools\termlite\termLite.exe	The TermLite executable
visiontools\termlite\setup.exe	The TermLite installation program

11.7 Answers to Review Questions

This section provides suggested 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 is to run TermLite 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 is to run TermLite 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. Two 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**(XC) or **scoterm**(XC), which could in turn be used to run **vi**(C).)

4. The strategies for providing word processors to Windows users can include:

   - Using a native Windows word processor such as Microsoft Word or Corel WordPerfect, with all data files stored on a UNIX file server and printing via shared UNIX printers. While this allows central management of file and printers, it involves installation and support of a word processor on every client machine. However, if the network or file server is down, the applications can still be used locally.

   - Using a X Window based word processor that runs on the UNIX system, such as WordPerfect for UNIX, Star Division's StarOffice, or Applix's ApplixWare. An X Window server for Microsoft Windows, such as XVision, could be used to use the word processor from Windows. Tarantella could also be used to provide client access. Documents and printers are still managed centrally, and now so is the application software. The X Window server can be used to run multiple applications (spreadsheets and graphing programs, for example), without additional client side software installation.

   - Using a Java based custom client agent, such as Applix AnyWare or StarOffice for Java. This would require a Java virtual machine on the client, as is found in most web browsers. This would probably create much less network traffic than the X Window approach. If users already have web browsers (as many do), then no client software installation is required.

5. The **Address Allocation Server** provides dynamic IP addresses from address pools for DHCP.

6. 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

Instructor Guide

Lesson 12

Mail Configuration

Suggested lesson duration

This lesson should take approximately 2.5 hours to complete.

Pre-lesson classroom preparation

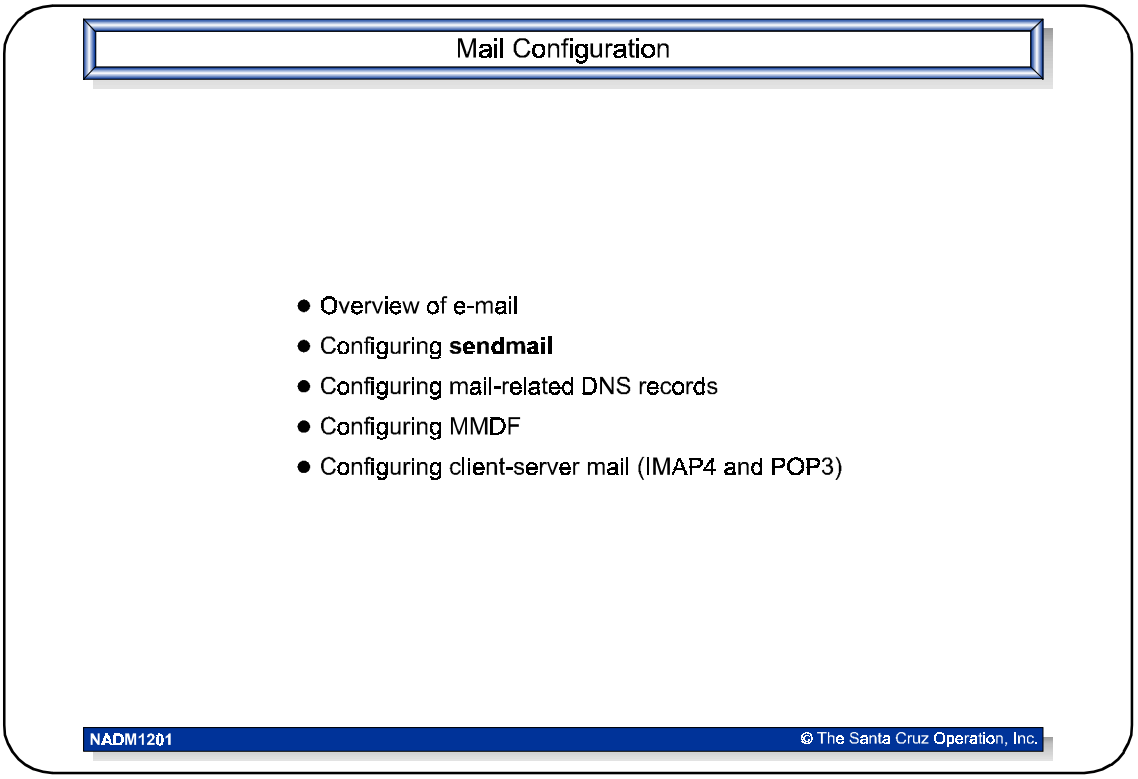
Have a working classroom network. You should have **sendmail**(ADMN) installed on server systems and one or more machines running a POP3 or IMAP4 mail client.

Instructor preparation

You must be fully familiar with the configuration of **sendmail**(ADMN) through the use of **mkdev cf** and the **Internet Manager**, as well as the underlying files and daemons that support **sendmail**, specifically the *sendmail.cf(SFF)* file. You should also be familiar with **mmdf**(ADM) and its configuration files. Good references for **sendmail**(ADM) include the O'Reilly & Associates Nutshell Handbook® and SCO documentation on the topic:

- Costales, Bryan with Eric Allman, ***sendmail*** Second Edition, Sebastapol, CA: O'Reilly & Associates, 1997, ISBN 1-56592-222-0
- **scohelp ⇨ Mail and Messaging**

AV 12-1



Instructor Guide

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 your users will use, namely e-mail.

SCO provides a complete e-mail environment, from user interface to transfer protocols, with many options that you can configure. This lesson describes the various components of the SCO e-mail environment and how you can configure them for the users of your network.

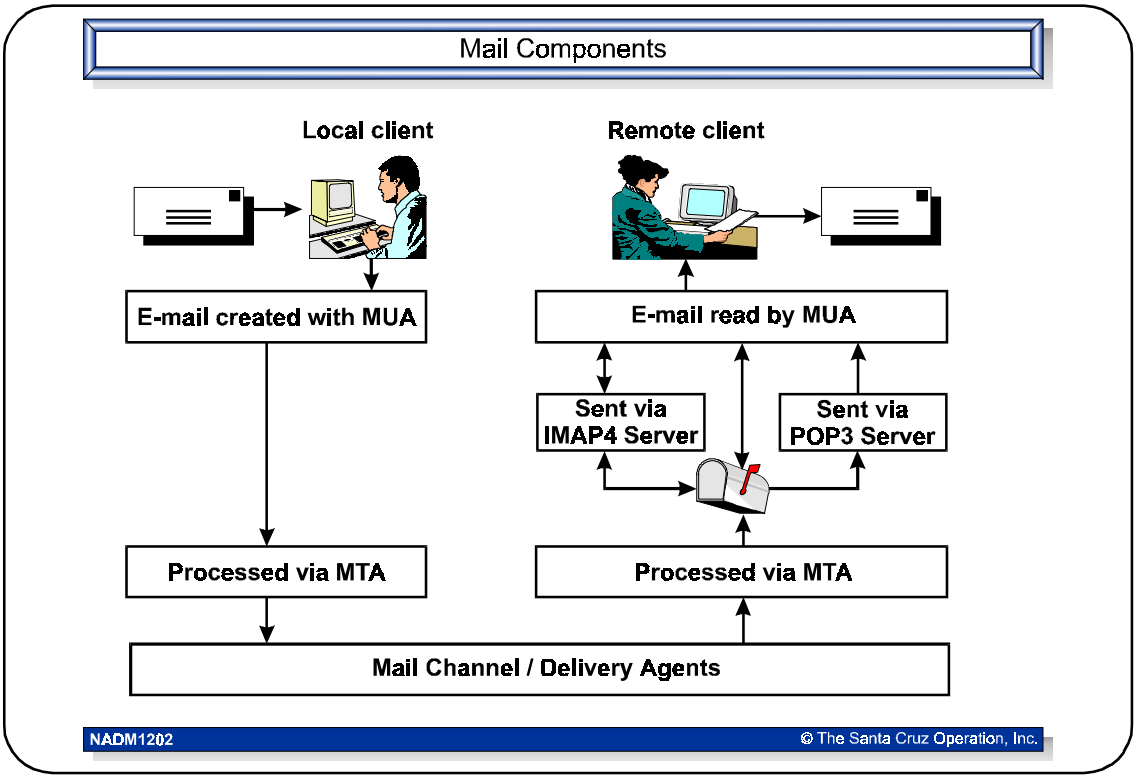
Objectives

This lesson describes the SCO e-mail environment. By the end of this lesson, you will be able to:

- List the components of an e-mail system
- List the mail delivery and transport agents provided with SCO OpenServer Release 5
- Configure **sendmail**(ADM)
- Create global aliases and mailing lists
- Modify the name server for use with a mail transport agent
- Describe how POP3 and IMAP4 services work
- Describe the basic steps to configure MMDF

AV 12-1: Mail Configuration
This first slide describes the objectives covered in this lesson.

AV 12-2


Mail Components
Local client
Remote client
E-mail created with MUA
E-mail read by MUA
Sent via IMAP4 Server
Sent via POP3 Server
Processed via MTA
Processed via MTA
Mail Channel / Delivery Agents
NADM1202
© The Santa Cruz Operation, Inc.

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: e-mail Components
Use this slide to talk about the various components of a modern e-mail system. It is analogous to a protocol stack, where you can replace any standards-conforming piece into any one of the layers.

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, SCO OpenServer Release 5 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.

NOTE: POP3 is a one-way protocol. It allows MUAs to transfer messages from the mail server MTA, as shown in the right-hand side of the slide. The slide shows a single arrow going from the MUA to the MTA on the left-hand side. In fact, if the message is being written by a POP3 client, the message is sent to the MTA on the server via SMTP, not POP3. IMAP4 is bi-directional, but outgoing e-mail is still sent by SMTP.

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, etc. These programs are often referred to as *helper programs*.

Discuss how you use MIME in your everyday e-mail tasks and why it is a desirable feature.

MUAs that come with SCO OpenServer Release 5 include:

mail(C) The traditional command-line mail. The SCO OpenServer Release 5 implementation of **mail** supports MIME and IMAP4.

scomail(XC) An X Window System™ mail program that is included with the OpenServer graphical desktop (**xdt3**(XC)). It is also MIME capable.

Netscape The web browser that is included with SCO OpenServer. It also includes a graphical e-mail program with MIME, POP, and HTML capabilities.

Instructor Guide

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 SCO OpenServer Release 5.

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 Pegasys. These MUAs will work with SCO OpenServer Release 5 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.

There are two MTA's provided with SCO OpenServer Release 5, **sendmail**(ADM) and **mmdf**(ADM)**.** Although MMDF is installed unless **sendmail** is specifically selected during installation, **sendmail** is the primary focus of this lesson**.** It is a very popular MTA that is available for most versions of UNIX and has gained widespread acceptance as the MTA for UNIX systems.

There are several other MTAs available, some via the Internet. One popular MTA is **smail**. It is based on **sendmail**, but is easier to configure.

The Netscape Mail Server™ is available as an optional MTA for SCO OpenServer Release 5. It contains many advanced features including:

- An easy to use interface
- 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)

12.1.3 Mail delivery agent

When the MTA has determined a route for delivery of a message, it calls the appropriate delivery program, or *agent,* to deliver the mail. The SCO OpenServer implementation of **sendmail** is configured to route mail to one of several delivery agents, also known as *mailers.*

The mailers available in the standard **sendmail** configuration include:

Local	Handles delivery of mail to local users by executing **lmail**(ADMN)
TCP	Delivers mail over TCP/IP networks using DNS and the Simple Mail Transport Protocol (SMTP).
UUCP	Uses **uux**(C) to deliver mail to hosts configured as UUCP connections in the */usr/lib/uucp/Systems* file.
Badhost	Queues messages that temporarily fail because the name server is not available. This can also be configured to forward e-mail for unknown hosts to a *smarter* mail host for delivery.
Baduser	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.
Multihome	Used for routing mail for virtual domains. Virtual domains allows a single host to manage mail for multiple domain names.
X.400	Delivers mail with X.400-style addresses to an X.400 gateway. X.400 is an MTA for the OSI networking protocol.

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 (i.e., 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

Make sure that students understand the difference between POP and IMAP. Both types of clients maintain a store on the host on which they are running. IMAP clients, when on-line, synchronize the client database with the server and make modifications to the server side as well as the client. POP only downloads messages from the server and may remove messages from the server as they are downloaded.

SCO OpenServer Release 5 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**(ADMN) 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 file */usr/lib/sendmail.cf*. This file stores information about the server and mailers
Aliases	Aliases used by **sendmail** to process mail are maintained in files located in */usr/lib/mail/aliases.*
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.

Each component is discussed in more detail in the next sections.

12.2.1 The default sendmail configuration

On SCO OpenServer Release 5, **sendmail** is preconfigured to support mail delivery on the local system and the LAN. For many situations, no additional **sendmail** configuration is required. Mail 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 **mail**(C) or **scomail**(XC) to send a message between the two users. Verify that it is delivered.

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 on a remote classroom system.

12.2.2 Modifying the sendmail configuration

Change the default **sendmail** configuration:

- To use a central mail server and you need to forward all mail to that server
- To configure this server as a central mail server
- To use other mail channels

All configuration options for **sendmail** are set in the *sendmail.cf*(SFF) file.

If you need to change the default **sendmail** configuration, there are three ways to modify the *sendmail.cf* file. These methods are not compatible—you must decide which tool to use based on the options you need to modify:

- **Internet Manager**—A graphical interface for configuring a limited set of options including:

Host hiding	The domain name that appears in the “From:” line of the message; known as *host hiding*
Forward local	Forwards mail with no @ in the “To:” line to a central system for processing
Forward non-local	Forwards mail with @ in the “To:” line to a central system for processing
Postmaster	Who receives messages reporting mail problems
Hostmaster	Who receives messages that report DNS problems

  The advantage of using the **Internet Manager** is simplicity. The disadvantage is that it configures only a limited set of options

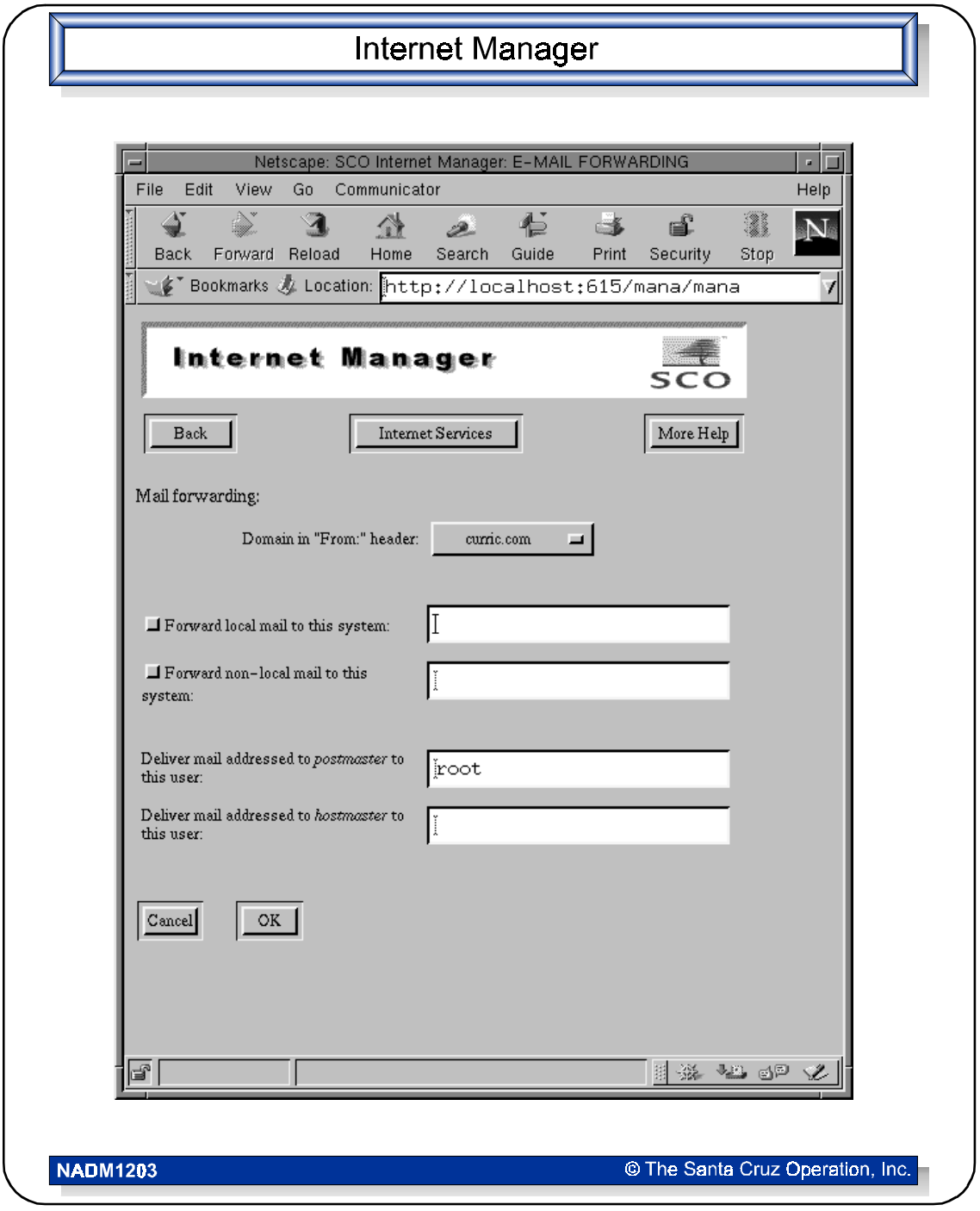
- **mkdev cf**—A character interface used to configure:

UUCP	Enables a UUCP channel
NIS maps	Provides for using NIS maps to define mail aliases
X.400	Forwards messages with X.400-format addresses to an X.400 gateway

- **Manual**—Editing the */usr/lib/sendmail.cf* file gives you complete control over all **sendmail** functionality. The disadvantage is that it is complex, laborious, and prone to error.

  WARNING: If you modify */usr/lib/sendmail.cf*, your changes are overwritten if you later use **mkdev cf**.

AV 12-3


Internet Manager
Netscape: SCO Internet Manager: E-MAIL FORWARDING
File Edit View Go Communicator Help
Back Forward Reload Home Search Guide Print Security Stop
Bookmarks Location: http://localhost:615/mana/mana
Internet Manager
SCO
Back
Internet Services
More Help
Mail forwarding:
Domain in "From:" header: curric.com
Forward local mail to this system:
Forward non-local mail to this system:
Deliver mail addressed to postmaster to this user:
root
Deliver mail addressed to hostmaster to this user:
Cancel
OK
NADM1203
© The Santa Cruz Operation, Inc.

12.2.2.1 Using the Internet Manager to configure sendmail

AV 12-3: Internet Manager
This shows the **Internet Manager sendmail**(ADMN) configuration window.

When the **Internet Manager** is used to configure **sendmail**, changes are made to */usr/lib/sendmail.cf*.

If you use the **Internet Manager** to configure **sendmail**, DO NOT use **mkdev cf** later. If you do, the changes made by the **Internet Manager** are lost. However, you can edit */usr/lib/sendmail.cf* later to manually make changes. See Section 12.2.2.3, “Manually configuring sendmail,” on page 12-19.

The problem with mkdev cf is explained in the next section.

To configure the **sendmail** forwarding and host hiding options:

Step 1. Start the **Internet Manager**.

Step 2. Enter account name **admin** and the **root** password.

Step 3. If virtual domains are configured, select “SYSTEM-WIDE” as the interface to configure.

Step 4. Click on the **Mail** icon.

Step 5. Fill in the form shown in AV 12-3 using these guidelines:

Host hiding	To change the domain name that appears in the “From:” line of the message, click on the button showing the fully-qualified domain name. The other choice is the domain name only. Selecting the domain name hides the host name.
Forward local	To forward all local mail (messages with no @ in the “To:” line) to a central system, toggle the field and enter the host name of the system to which all local mail is forwarded.
Forward non-local	To forward all non-local mail (messages with an @ in the “To:” line) to a central system, toggle the field and enter the host name of the system to which all local mail is forwarded.

Postmaster	Change the user name if an account other than root receives messages reporting mail problems
Hostmaster	Enter a user name for reporting DNS problems if this system is a primary name server.

Exercise: Configuring host hiding

Use the **Internet Manager** to hide your host name.

Send mail between two user accounts and note the domain name in the "From:" line.

12.2.2.2 Using mkdev cf to configure sendmail

The command **mkdev cf** is a menu system that:

- Changes a set of configuration parameters, such as UUCP and X.400 connection information, and NIS support
- Generates a new *sendmail.cf* file
- Restarts the **sendmail** daemon

The new *sendmail.cf* file is created by combining the changes entered through menu selections—stored in */usr/lib/sendmail.d/sendmail.ans*—with the defaults contained in the file */usr/lib/sendmail.d/sendmail.srf*. The program **gencf** uses these changes to generate a new *sendmail.cf* file

The original *sendmail.cf* file is overwritten when **mkdev cf** calls **gencf** to generate the new *sendmail.cf* file. As discussed in the previous section, the **Internet Manager** modifies the *sendmail.cf* file directly. This is why any changes previously made by the **Internet Manager** are lost when **mkdev cf** is used.

For more information about using **mkdev cf** to configure **sendmail**, see **scohelp ⇨ Mail and Messaging Guide ⇨ Chapter 5: sendmail administration ⇨ Standard sendmail configuration ⇨ Running mkdev cf**.

12.2.2.3 Manually configuring sendmail

You might need to change **sendmail** configuration parameters that are not accessible using **mkdev cf** or the **Internet Manager**. These are examples of features that can be enabled by manually editing the *sendmail.cf* file:

- Managing unsolicited email (spam)
- Changing the interval for using 'expensive' connections to deliver mail
- Altering message timeouts so undeliverable mail is returned more quickly
- Setting the delivery mode to specify how quickly mail is delivered

You might also want to replicate a given **sendmail** configuration to multiple machines by copying the *sendmail.cf* file from one system to another. At a minimum, the host name needs to be changed for each replicated system.

If you manually make changes to */usr/lib/sendmail.cf,* you can later use the **Internet Manager**, since it also directly modifies */usr/lib/sendmail.cf.*

WARNING: You cannot use **mkdev cf** after editing */usr/lib/sendmail.cf*. It overwrites the existing *sendmail.cf* file and any changes made directly to the file are lost.

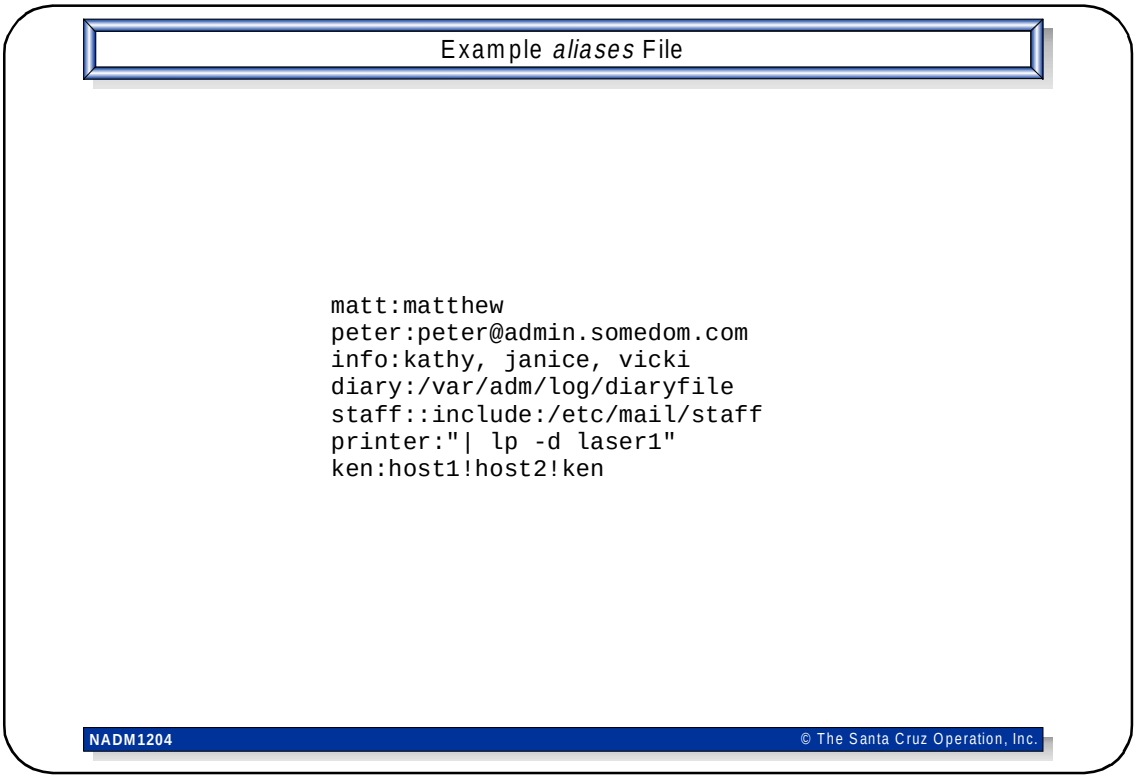
Another alternative is to edit the source file used by **mkdev cf**, */usr/lib/sendmail.d/sendmail.srf.* Then **gencf**, or **mkdev cf**, can be used to generate the new *sendmail.cf* file. This prevents you from overwriting manual changes. However, it is not documented in SCO documentation and might not be supported by SCO.

Manually configuring **sendmail** requires a thorough knowledge of both SMTP and **sendmail**. For more information regarding the structure and modification of the *sendmail.cf* file, read:

Costales, Bryan with Eric Allman, ***sendmail*** Second Edition, Sebastapol, CA: O'Reilly & Associates, 1997, ISBN 1-56592-222-0

The Web site, *http://www.sendmail.org,* is another good reference.

AV 12-4


Example *aliases* File
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
NADM1204
© The Santa Cruz Operation, Inc.

12.2.3 Configuring mail 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

AV 12-4: Example *aliases* File
This slide shows an example *aliases*(SFF) file.

By default, aliases are configured in the */usr/lib/mail/aliases* file. This *aliases*(SFF) file can be modified using a text editor to create entries in this format:

`aliasname:`*`address`*`[, address]`

where *address* is:

- 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. For example, the address for an alias named `staff` can be:

  ```
  :include:/etc/mail/staff
  ```

- A command to which messages are piped. For example, the address for an alias named `printer` can be:

  ```
  "| lp -d laser1"
  ```

- A UUCP e-mail address. For example *host1!host2!user*

An example *aliases* file is shown in AV 12-4.

After editing the */usr/lib/mail/aliases* file, you must run the **newaliases**(ADMN) command to rebuild the **sendmail** alias database, */usr/lib/mail/aliases.db*. Enter:

```
newaliases
```

If you receive error messages that */usr/lib/mail/aliases* is an unsafe map file, it is due to a problem using **newaliases** with symbolic links. To work around this: (1) rename the files *aliases* and *aliases.db* (to any new name); (2) recreate the files with a text editor; (3) rerun **newaliases**.

Exercise: Creating mail aliases

Step 1.	Edit */usr/lib/mail/aliases* and add a new alias for two or more of your local users.
Step 2.	Add an alias for users on another classroom system.
Step 3.	Run **newaliases**(ADMN) to update */usr/lib/mail/aliases.db*.
Step 4.	Send mail to the aliases and verify that it is received.

Instructor Guide

The mail system can be configured to use a variety of alias files, each using a different database format. For example, NIS *maps* can be used as alias files. For more information about using NIS maps and other alias files, see **scohelp ⇨ Mail and Messaging Guide ⇨ Chapter 5: sendmail administration ⇨ Administering sendmail ⇨ Maintaining the alias database.**

12.2.4 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 (**vacation**(C) mail)

12.2.5 *.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 to go to **donna** on *machine2.somedom.com*.

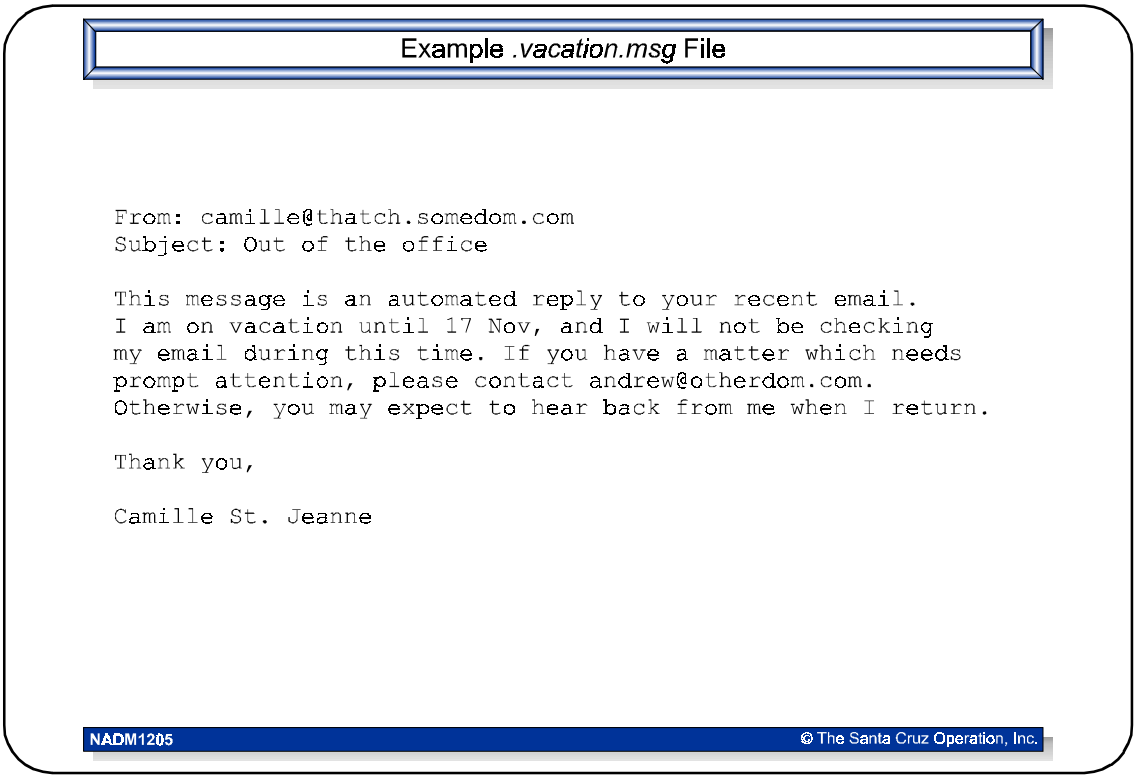
You should be careful not to configure two machines to redirect mail to each other, as this creates an endless mail loop.

12.2.5.1 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.

This can be configured using the **vacation**(C) program from the command line. Two files in the users home directory control the behavior of vacation notification, *.vacation.msg* and *.vacation.db*. The **vacation** command is usually used in conjunction with a *.forward* file.

AV 12-5


Example .vacation.msg File
From: camille@thatch.somedom.com
Subject: Out of the office
This message is an automated reply to your recent email.
I am on vacation until 17 Nov, and I will not be checking
my email during this time. If you have a matter which needs
prompt attention, please contact andrew@otherdom.com.
Otherwise, you may expect to hear back from me when I return.
Thank you,
Camille St. Jeanne
NADM1205
© The Santa Cruz Operation, Inc.

To set up extended absence notification:

Step 1. Create the *.vacation.msg* file in your home directory. It must include the "From:" line and can include other mail header lines, as shown in AV 12-5.

Step 2. Initialize the *.vacation.db* file with the command line:

```
vacation -i
```

Step 3. Create a *.forward* file, substituting your user name as appropriate:

```
\camille, "|vacation camille"
```

Note the backslash (\) before the user name. This prevents mail from infinitely looping through the *.forward* file.

Step 4. Set permissions to allow everyone to read *.forward*:

```
chmod a+r .forward
```

Step 5. Set permissions to allow everyone to execute your home directory. For example:

```
chmod a+x /usr/camille
```

Permissions are a problem with later versions of **sendmail** that do not run as **root** for security reasons.

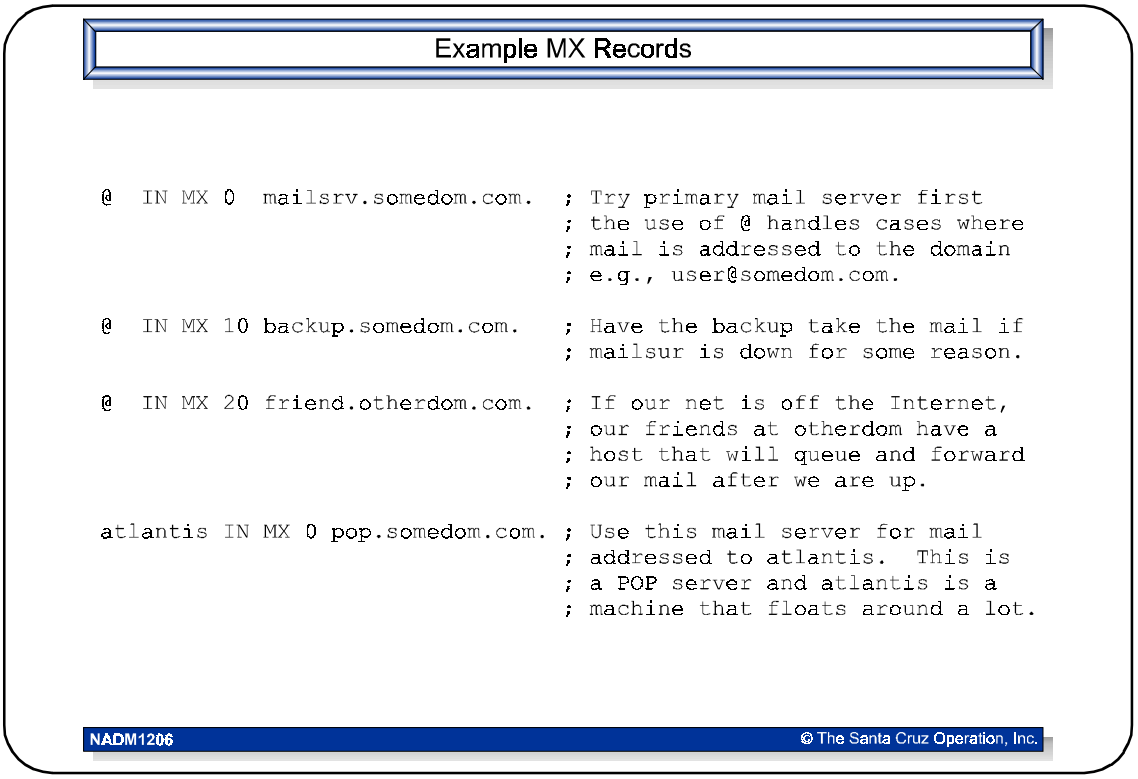
AV 12-5: Example *.vacation.msg* File
This slide shows an example *.vacation.msg* file used with the **vacation**(C) command to reply to messages automatically.

Exercise: Using vacation(C) mail

Set up **vacation**(C) mail for a user account on your system using the steps above.

Test the configuration by sending mail from another user account.

AV 12-6


Example MX Records
@ IN MX 0 mailsrv.somedom.com. ; Try primary mail server first
; the use of @ handles cases where
; mail is addressed to the domain
; e.g., user@somedom.com.
@ IN MX 10 backup.somedom.com. ; Have the backup take the mail if
; mailsur is down for some reason.
@ IN MX 20 friend.otherdom.com. ; If our net is off the Internet,
; our friends at otherdom have a
; host that will queue and forward
; our mail after we are up.
atlantis IN MX 0 pop.somedom.com. ; Use this mail server for mail
; addressed to atlantis. This is
; a POP server and atlantis is a
; machine that floats around a lot.
NADM1206
© The Santa Cruz Operation, Inc.

12.3 Configuring DNS for use with mail

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.

MX records have the format:

```
Name        IN    MX    Priority  Host
```

where:

Name — The machine to which the mail is destined. This can also be the name of your domain. If that is your intent, you can also use the @ symbol, which in DNS means *the current domain.*

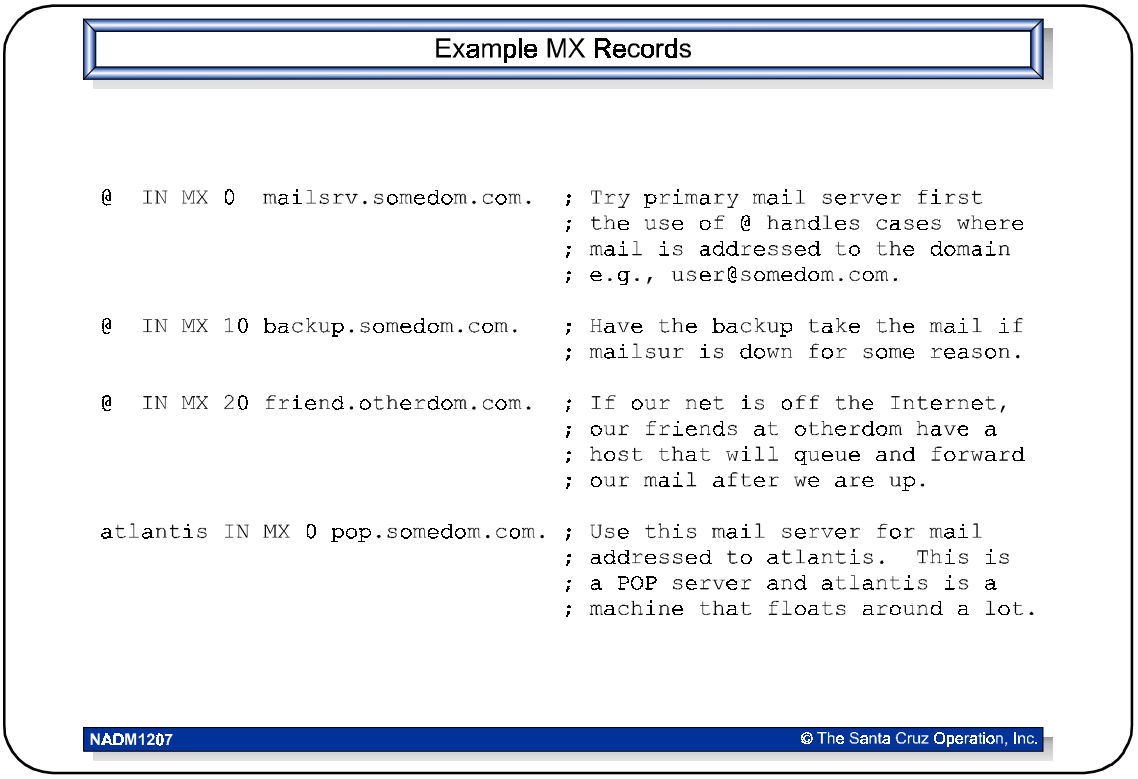
Priority — The relative priority that is assigned to this mail exchanger. The highest priority is 0 and the lowest is 32767. The size of this number does not matter, only its relation to the other MX servers.

Your mail server should be set with the highest priority (usually 0) so all mail messages are eventually sent to it.

Host — The name of the host that should receive the mail. In the case of mail exchangers outside of your domain, they will just queue your mail for you until your mail exchanger becomes available again.

AV 12-6: Example MX Records
Use this slide to illustrate the use of MX records. Walk students through each line in the example.

AV 12-7


Example MX Records
@ IN MX 0 mailsrv.somedom.com. ; Try primary mail server first
; the use of @ handles cases where
; mail is addressed to the domain
; e.g., user@somedom.com.
@ IN MX 10 backup.somedom.com. ; Have the backup take the mail if
; mailsur is down for some reason.
@ IN MX 20 friend.otherdom.com. ; If our net is off the Internet,
; our friends at otherdom have a
; host that will queue and forward
; our mail after we are up.
atlantis IN MX 0 pop.somedom.com. ; Use this mail server for mail
; addressed to atlantis. This is
; a POP server and atlantis is a
; machine that floats around a lot.
NADM1207
© The Santa Cruz Operation, Inc.

Another use for MX records is to reroute mail for a hostname to another host. The first hostname 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 hostname 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 you can:

AV 12-7: Example MX Records
This slide is repeated for this example.

- Create MX records for the name *somedom.com*. This is shown in AV 12-7. The MX records are usually included in the *named.hosts* file.
- 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 Lesson 8: “Domain Name System (DNS) Configuration” for general information on DNS configuration.

12.4 Enabling and Disabling 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**(ADM) when **inetd** detects POP3 or IMAP4 requests.

Using these services requires no configuration on the server.

To disable these servers, edit */etc/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, signal the **inetd** process to re-read this file. Enter:

```
kill -1 ‘cat /etc/inetd.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.

12.5 MMDF

The MMDF MTA has several benefits that are not provided by **sendmail**:

- Easier configuration—a more complete set of configuration tools and human readable configuration files
- Additional user-configuration features
- A larger set of supported delivery agents

As with **sendmail**, MMDF is preconfigured to support local and SMTP mail delivery as follows:

Incoming	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	Delivered directly to the system in the address (this might be the local system)
	Append the fully-qualified domain name to the user name in the "From:" line

For many situations, no additional MMDF configuration is required. The next sections provide a brief overview of MMDF configuration. For detailed information, see **scohelp ⇨ Mail and Messaging Guide ⇨ Chapter 4: Managing mail with MMDF.**

12.5.1 MMDF configuration

MMDF is configured using either **mkdev mmdf** or the **Mail Configuration Manager**. Unlike **sendmail**, these tools are compatible and either can be used to perform MMDF configuration.

In addition to configuring delivery channels, forwarding, and host hiding, the **Mail Configuration Manager** can also be used to create aliases, specify mailbox locations, and configure its host table when DNS is not used for name resolution.

Although the **Mail Configuration Manager** provides access to many MMDF configuration options, it does not include all features and some manual configuration might be necessary.

MMDF uses several mail configuration files:

mmdftailor(F)	Site-dependent configuration information; this is the main configuration file, which references other supporting files
**.dom*	Tables used to transform mail addresses into standard format; see *tables*(F)
**.chn*	Tables used to associate a machine name with a delivery program
*alias.**	Tables containing alias configuration information

The user account, **mmdf**, is created when MMDF is installed. It is used to manually configure these files, which are located in the **mmdf** home directory, */usr/mmdf*.

The basic approach for manually configuring MMDF is:

Step 1. Use the **Mail Configuration Manager** to do as much configuration as possible.

Step 2. Log in as **mmdf.**

Step 3. Edit the MMDF configuration files in */usr/mmdf* as described in the *mmdftailor*(F) and *tables*(F) manual pages.

Step 4. Rebuild the hashed database using **dbmbuild**(ADM).

Step 5. Restart the **deliver**(ADM) daemon.

12.5.2 MMDF user-configurable features

As with **sendmail**, MMDF provides user-configurable features such as forwarding and automatic replies during extended absences. In addition, MMDF provides a more powerful and flexible means of processing incoming mail.

All user configuration of incoming mail is handled by the *.maildelivery*(F) file in the user's home directory

12.5.2.1 The .maildelivery file

Your inbound mail can be processed according to a complex set of selection criteria, specified in the *.maildelivery* file. 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
```

For example, this line causes 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
```

As another example, this line takes 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*(F) manual page.

You might mention how. *maildelivery* can be useful for blocking spam.

12.6 Lesson Summary

This lesson described the components of a full e-mail environment and presented the information necessary for you to configure that mail environment, including making modifications to your DNS databases. Configuration of the **sendmail** MTA, and an overview of MMDF configuration were presented.

Review questions

1. What are the three layers in an e-mail system?

2. Which mail delivery agents are configured by default on SCO OpenServer?

3. What do you have to do initially to enable e-mail on your SCO OpenServer system?

4. Which records should you add to the DNS database to tell the Internet which hosts act as mail exchangers for your network?

5. What must be done to the *inetd.conf* file to disable POP3 services under SCO OpenServer Release 5?

12.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

dbmbuild(ADM)	Command to rebuild the MMDF *dbm* databases
deliver(ADM)	MMDF mail delivery program
imapd(ADMN)	Daemon that handles IMAP mail access
mail(C)	Traditional UNIX command-line mail program
mkdev cf(ADM)	Command to configure the **sendmail** MTA
mkdev mmdf(ADM)	Command to configures the MMDF MTA
newaliases(ADM)	Command to rebuild the **sendmail** aliases file
popper	Daemon that handles POP mail access
scomail(XC)	X Window System mail program included in CDE
sendmail(ADM)	Standard SCO OpenServer mail transfer program

Relevant files

/etc/default/mail	Global mail configuration file
/etc/inetd.conf	Internet services configuration file
/etc/mail/slocal	Program that performs local mail delivery
/etc/mail/multihome	Program that routes mail for virtual domains to the correct user account
$HOME/.forward	File which redirects mail to the address listed in the file
$HOME/.maildelivery	File which configure automatic handling of incoming mail
$HOME/.vacation.msg	File which specifies a message to be sent using the **vacation**(C) command

$HOME/.vacation.db	Database containing the list of mail addresses sent automatic replies using **vacation**(C)
/usr/lib/sendmail.cf	Configuration file for **sendmail**(ADM)
/usr/lib/mail/aliases	Default location for system mail aliases
/usr/lib/uucp/uucico	Unix-to-Unix copy system is used to transfer mail for UUCP channels
/usr/mmdftailor	Master MMDF configuration file
/usr/mmdf/table/.chan*	MMDF channel files
/usr/mmdf/table/.dom*	MMDF domain files
*/usr/mmdf/table/alias.**	MMDF alias files

12.8 Answers to Review Questions

This section provides suggested 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. SCO OpenServer provides the **sendmail** and MMDF transport agents. Each is configured by default to use the local and SMTP delivery agents.

3. Nothing needs to be done to enable e-mail. Once installed and connected to the network, SCO OpenServer is preconfigured to send and receive e-mail.

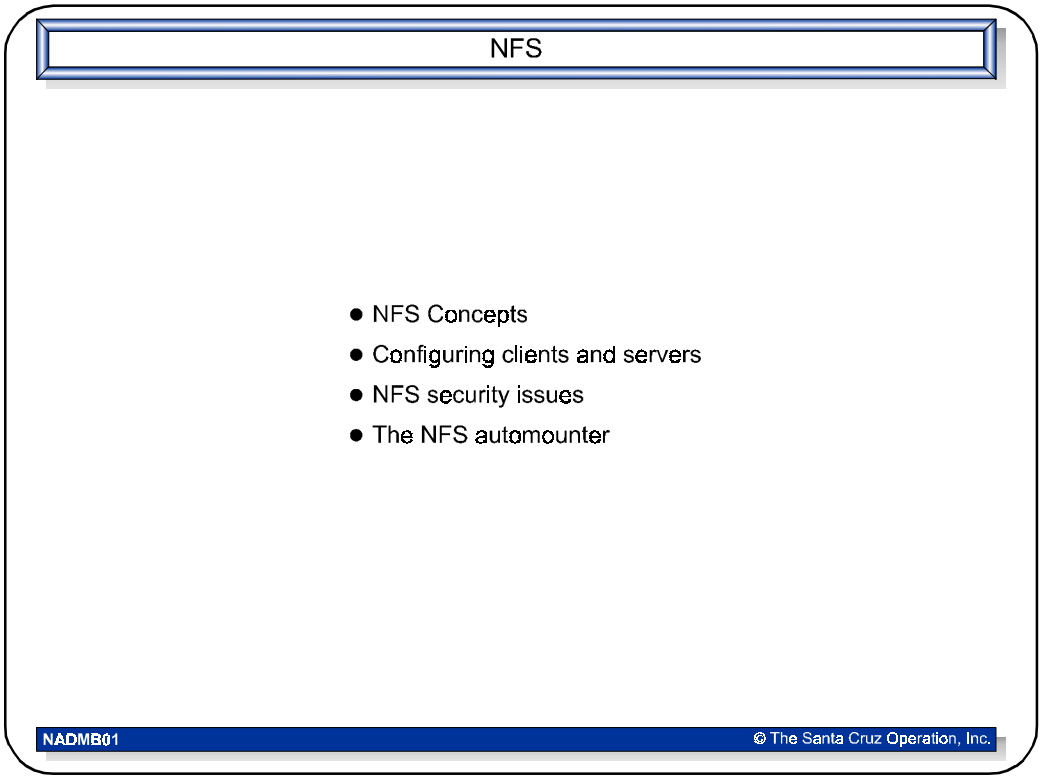
4. You should add `MX` records to the DNS database to tell the Internet which hosts act as mail exchangers for your network.

5. 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`**.

A

NFS

AV A-1


NFS
NFS Concepts
Configuring clients and servers
NFS security issues
The NFS automounter
NADMB01
© The Santa Cruz Operation, Inc.

Appendix A

NFS

Introduction

The Network File System (NFS®) is a tool that people use passively, often without realizing that they are using it. If you configure NFS properly, your job as a network administrator is easier, and your users are able to use disk space throughout your network transparently.

NFS allows you to make entire filesystems or directories on one host available to many other hosts that treat the disk space as though it were their own. NFS accomplishes this by allowing you to mount the remote disk resource much like you can mount a local filesystem.

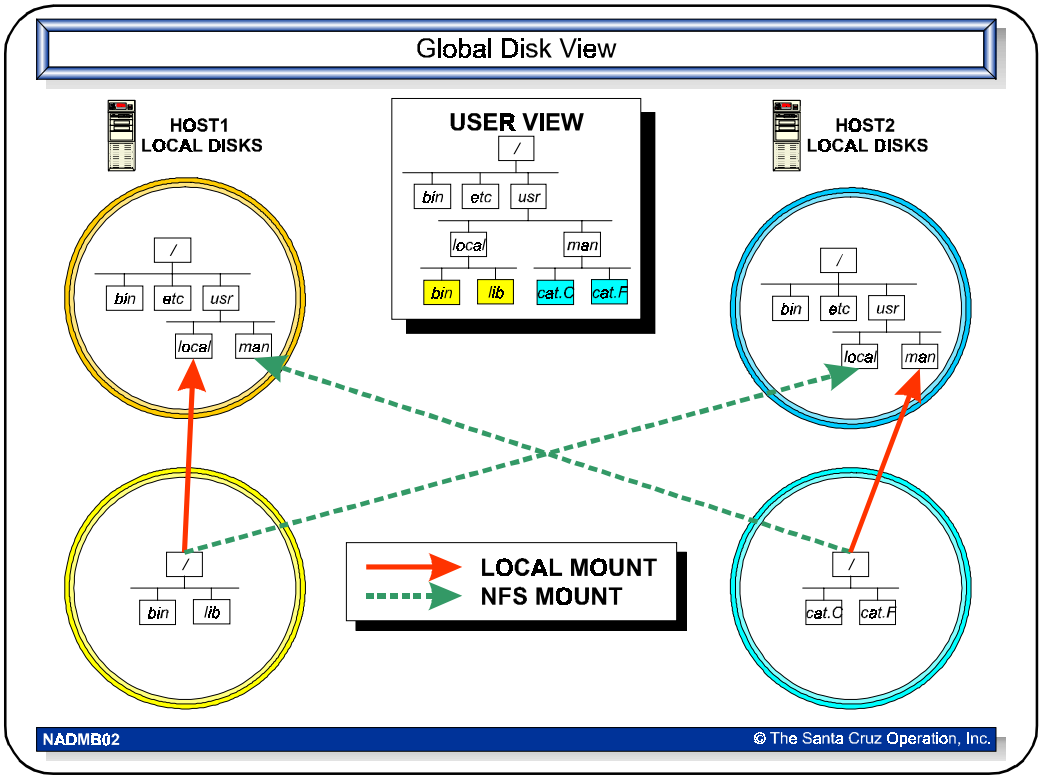
The automounter allows you to leave NFS resources unmounted until they are needed, conserving local and network resources.

Objectives

This appendix describes NFS. By the end of this appendix, you will be able to:

- Describe NFS
- Explain how to start and stop NFS
- Configure NFS servers and clients
- Describe NFS security issues
- Configure the NFS automounter
- Explain the functions of the map files

AV A-2


Global Disk View
HOST1
LOCAL DISKS
USER VIEW
HOST2
LOCAL DISKS
/
bin
etc
usr
local
man
lib
cat.C
cat.F
LOCAL MOUNT
NFS MOUNT
NADMB02
© The Santa Cruz Operation, Inc.

A.1 NFS Concepts

The Network File System:

- Allows you to treat filesystems or directories on remote hosts as though they were local filesystems and thus mountable, readable, and writable.
- Requires that all administration of remote filesystems (**fsck**(ADM) and **mkfs**(ADM)), for example) to be done on the system that actually has the physical disk.
- Allows mounting of NFS filesystems to your file tree hierarchy with the same command as local filesystems.

A.1.1 NFS terms

These are terms used with NFS:

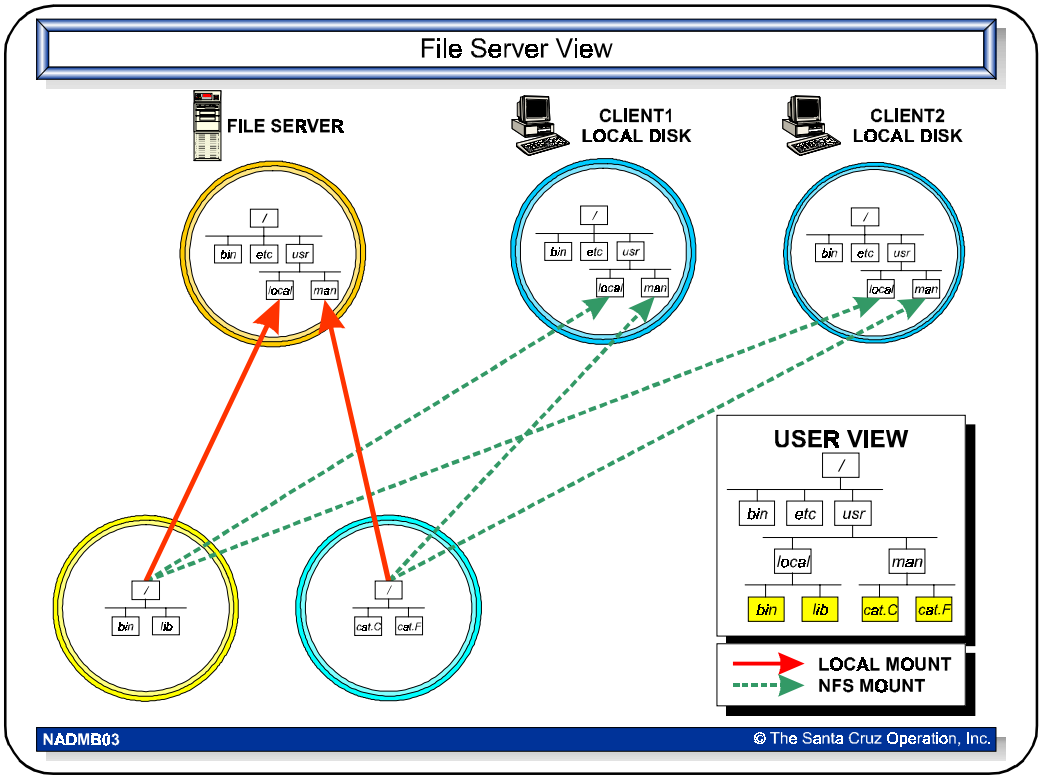
NFS server The host that has the physical disk and shares it using NFS.

NFS client The host that accesses the remote filesystem using NFS.

An NFS server can share its disk with many NFS clients. An NFS server sharing some of its disk might be an NFS client for another NFS server's disk.

AV A-2 and AV A-3 illustrate two example uses for NFS. The first example uses NFS to provide every host the same view of the network's disk space. The second example uses NFS as a file server for multiple workstations.

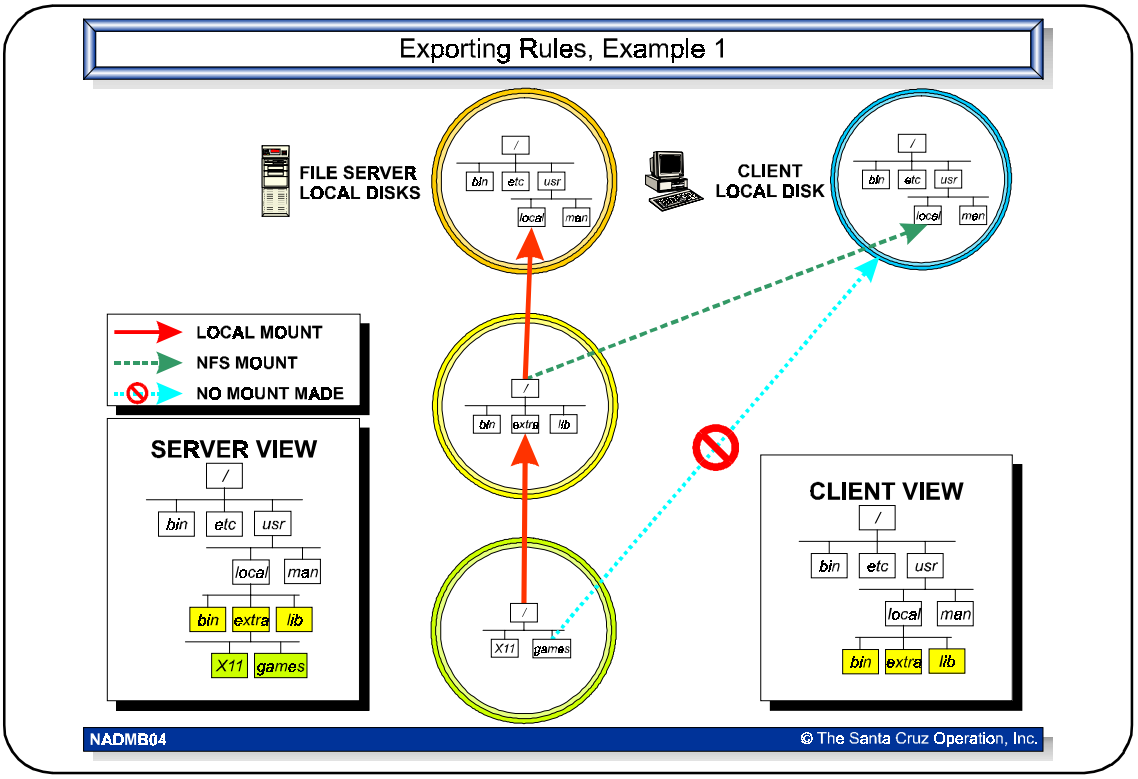
AV A-3


File Server View
FILE SERVER
CLIENT1
LOCAL DISK
CLIENT2
LOCAL DISK
/
bin
etc
usr
local
man
lib
cat.C
cat.F
USER VIEW
LOCAL MOUNT
NFS MOUNT
NADMB03
© The Santa Cruz Operation, Inc.

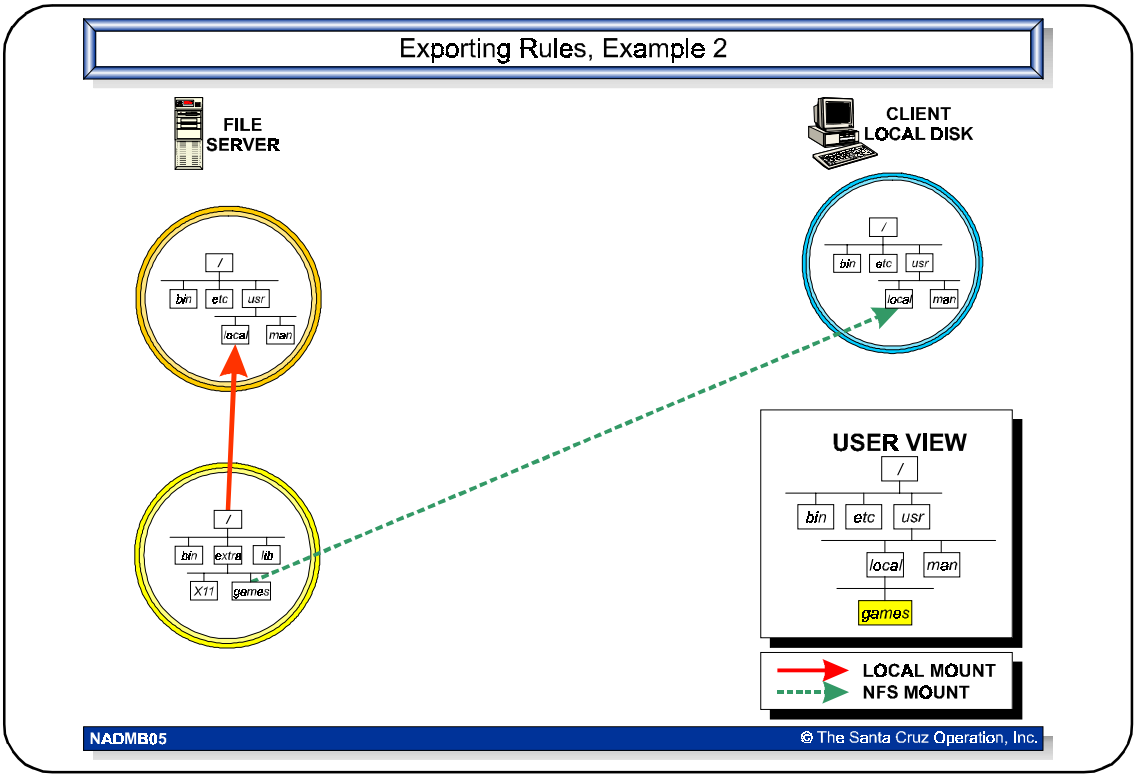
Remote resource	The directory that the server exports. It is often written as: *host*:*pathname* where host is the name of the file server and pathname is the absolute pathname on the server of the directory being exported.
Mount point	The directory where the remote resource is mounted onto the client.

In addition to this lesson, information about NFS and how it works can be found in **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Network File System (NFS)**.

AV A-4


Exporting Rules, Example 1
FILE SERVER
LOCAL DISKS
CLIENT
LOCAL DISK
LOCAL MOUNT
NFS MOUNT
NO MOUNT MADE
SERVER VIEW
CLIENT VIEW
/
bin
etc
usr
local
man
extra
lib
X11
games
NADMB04
© The Santa Cruz Operation, Inc.

AV A-5


Exporting Rules, Example 2
FILE
SERVER
CLIENT
LOCAL DISK
USER VIEW
/
bin
etc
usr
local
man
extra
lib
X11
games
LOCAL MOUNT
NFS MOUNT
NADMB05
© The Santa Cruz Operation, Inc.

A.1.2 Rules of importing and exporting

Note that in both of the previous slides, the arrows indicating NFS mounts, point to directories in the user's file tree, not to filesystems. This is because servers export directories, not filesystems.

If a server has exported a directory, */usr/local*, for example, and there is a filesystem mounted on */usr/local/extra*, then the contents of */usr/local/extra* are not visible to the clients unless:

- The server explicitly exports the */usr/local/extra* directory.
- The clients explicitly mounts the */usr/local/extra* directory.

This is illustrated in AV A-4.

Clients can import any subdirectory of an exported directory, provided that the subdirectory is not on another filesystem. This is illustrated in AV A-5.

A host cannot export a directory it has imported. For example, if a host mounts */usr/local* using NFS, it cannot export */usr/local* to another host.

A subdirectory of an already exported directory cannot be exported, unless the subdirectory is on a different filesystem. For example, you cannot export the root directory (*/*) as read-only and the */tmp* directory as read-write.

AV A-6


Filesystem Manager Export
Add Export Configuration on spider1
Directory to Export:
Select...
Read-only Clients:
None
Select...
Read-write Clients:
All systems
Select...
Export Now
Yes
No
Advanced Options
Anonymous Users...
OK
Cancel
Help
NADMB06
© The Santa Cruz Operation, Inc.

A.2 Configuring Servers

NFS server configuration can be done through the **Filesystem Manager** in **scoadmin**(ADM) or by configuring the files by hand and executing a few simple commands. As a rule, it is easier to use **scoadmin** to perform most system administration functions.

Configuring NFS file servers is a matter of listing which directories you want the server to export and to which clients they are to be exported.

A.2.1 Using scoadmin(ADM)

The **scoadmin**(ADM) **Filesystem Manager** can be used to export a filesystem or to modify the parameters of exported filesystems.

To export a new filesystem from **scoadmin**, select **Filesystems ⇨ Filesystem Manager ⇨ Export ⇨ NFS ⇨ Add Export Configuration**. This menu selection yields the form shown in AV A-6, which is filled out using these guidelines:

Directory to Share

Enter the name of the directory to share.

Alternately, click on **Select** to pop up a file-selection dialog box, allowing you to navigate through the file tree and find the directory to export.

Read-only and Read-write Clients

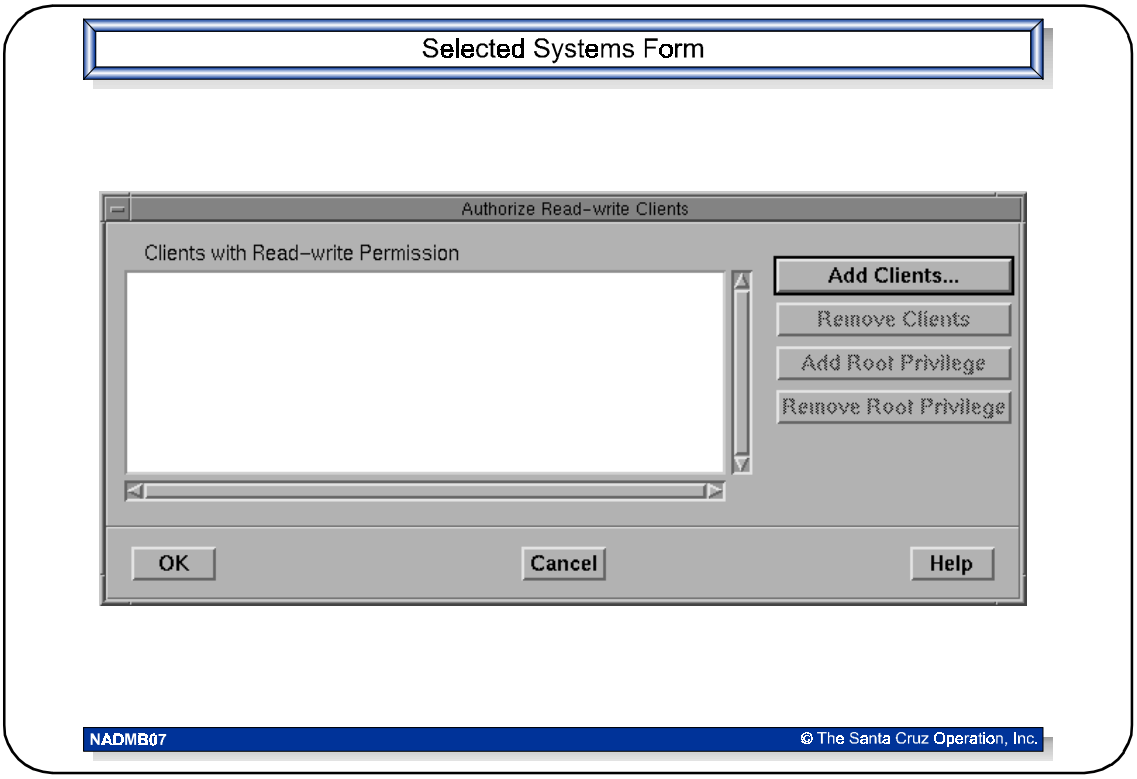
These two entries on the form work together to indicate whether the filesystem is being exported read-only or read-write (or even read-mostly).

If you set **Read-only Clients** to **All systems**, **Read-write** defaults to **None**. This means that the directory is exported read-only, and that any host can mount it. You can also set Read-write to Selected systems. This means that the directory is exported read-mostly. In other words, all hosts will be able to read and a few select hosts will be able to read and write.

If you set **Read-only Clients** to **Selected Systems**, **Read-write Clients** defaults to **None**. This means that the directory is exported read-only and only a selected set of hosts will be able to mount it.

If you set **Read-only Clients** to **None**, **Read-write Clients** defaults to **All**. It means any host will be able to mount it read-write.

AV A-7


Selected Systems Form
Authorize Read-write Clients
Clients with Read-write Permission
Add Clients...
Remove Clients
Add Root Privilege
Remove Root Privilege
OK
Cancel
Help
NADMB07
© The Santa Cruz Operation, Inc.

	If a directory is read-write accessible to a host, the host can mount the directory read-write or read-only. If a directory is read-only accessible to a host, the host can only mount the directory read-only. If you set either of the buttons to **Selected systems**, you can then click on **Select** to pop up the form that is displayed in AV A-7.
Export Now	The **Filesystem Manager** runs **exportfs**(NADM) now. Otherwise, the directory is not exported until the next time NFS is started.
Export at System Startup	The **Filesystem Manager** makes an entry in the */etc/exports* file. For further information see Section A.2.2.1, “The /etc/exports file,” on page A-14.
Anonymous Users	Indicate how anonymous UIDs and writes should be treated. Anonymous UIDs are those present on the client system that do not exist on the server system. See Section A.4, “NFS Security Issues,” on page A-30 for further information on UIDs and NFS.

This interface does not allow all the unique combinations you can get by editing the */etc/exports* file directly, but it is sufficient for most needs.

A.2.2 Configuring servers by hand

You can share a filesystem at any time by using the **exportfs**(NADM) command. However, the share is only in effect until NFS is shut down.

If you would like to always have a resource shared, you must take two steps:

Step 1. Modify the list of filesystems the NFS server is willing to share with other systems, and in what way it is willing to share them. In NFS, sharing is called *exporting*. This information is kept in the */etc/exports* file.

Step 2. Notify the **mountd**(NADM) daemon that the resource is available by executing the **exportfs**(NADM) command.

A.2.2.1 The */etc/exports* file

The */etc/exports* file contains a list of directories that can be exported to NFS clients. Each exported directory can be tailored in ways that allow:

- A restricted list of hosts to access a resource
- Any host to access a resource
- Read-only by certain hosts
- Read-write by certain hosts
- Super-user privileges by certain hosts

The format and meaning of the various tags in the file can be found in the **export**(NF) manual page. AV A-8 on page A-16 provides some example */etc/exports* entries.

AV A-8


Example /etc/exports File
/usr/man -ro,rw=pubadm
/usr/local/lib -access=host2,root=host2
NADMB08
© The Santa Cruz Operation, Inc.

The entries in AV A-8 are based on the slide AV A-2 on page A-4. These entries are for *host1* exporting its */usr/man* and */usr/local/lib* directories. In the first entry:

- *host2* has read-only access to the */usr/man* directory.
- Only users on the *pubadm* machine, not shown in the picture, can modify the contents.

In the second entry:

- *host2* has read-write access to the */usr/local/lib* directory.
- The root user on *host2* keeps its super-user privileges while accessing files on this NFS server. Normally, root user privileges do not cross NFS mounts. For further information about this, see Section A.4, “NFS Security Issues,” on page A-30.

A.2.2.2 The exportfs(NADM) command

The **mountd**(NADM) daemon reads the */etc/exports* file only when NFS is started. When changes are made to the */etc/exports* file after NFS has been started, the **exportfs**(NADM) command is used to notify **mountd** that the export list has changed.

To export all directories listed in */etc/exports*, enter:

```
exportfs -a
```

To stop exporting all directories listed in */etc/exports*, enter:

```
exportfs -u
```

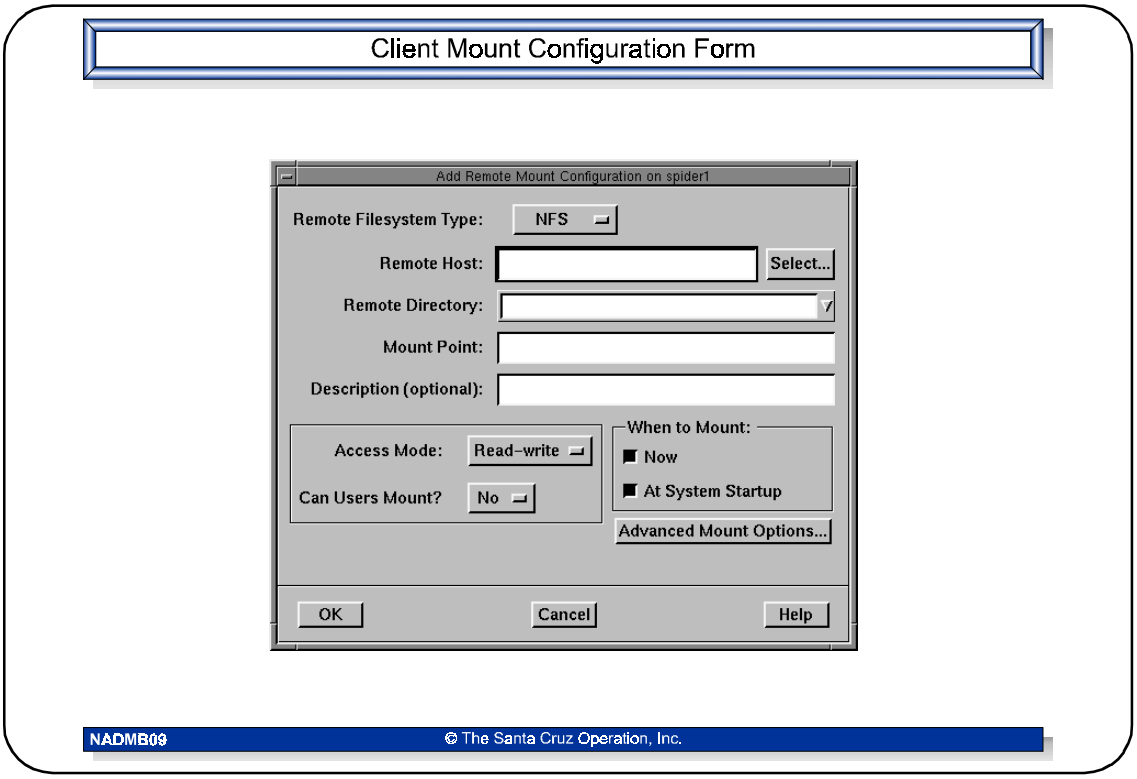
To stop exporting a specific directory, enter:

```
exportfs -u /usr/local
```

View exported directories using **exportfs** with no flags. Enter:

```
exportfs
```

AV A-9


Client Mount Configuration Form
Add Remote Mount Configuration on spider1
Remote Filesystem Type:
NFS
Remote Host:
Select...
Remote Directory:
Mount Point:
Description (optional):
Access Mode:
Read-write
Can Users Mount?
No
When to Mount:
Now
At System Startup
Advanced Mount Options...
OK
Cancel
Help
NADMB09
© The Santa Cruz Operation, Inc.

A.2.3 NFS client configuration

Configuring NFS clients is a matter of listing which directories you want the client to mount and from which server they should be mounted. As with server configuration, client configuration can be performed either using **scoadmin**(ADM) or using a text editor at the command line.

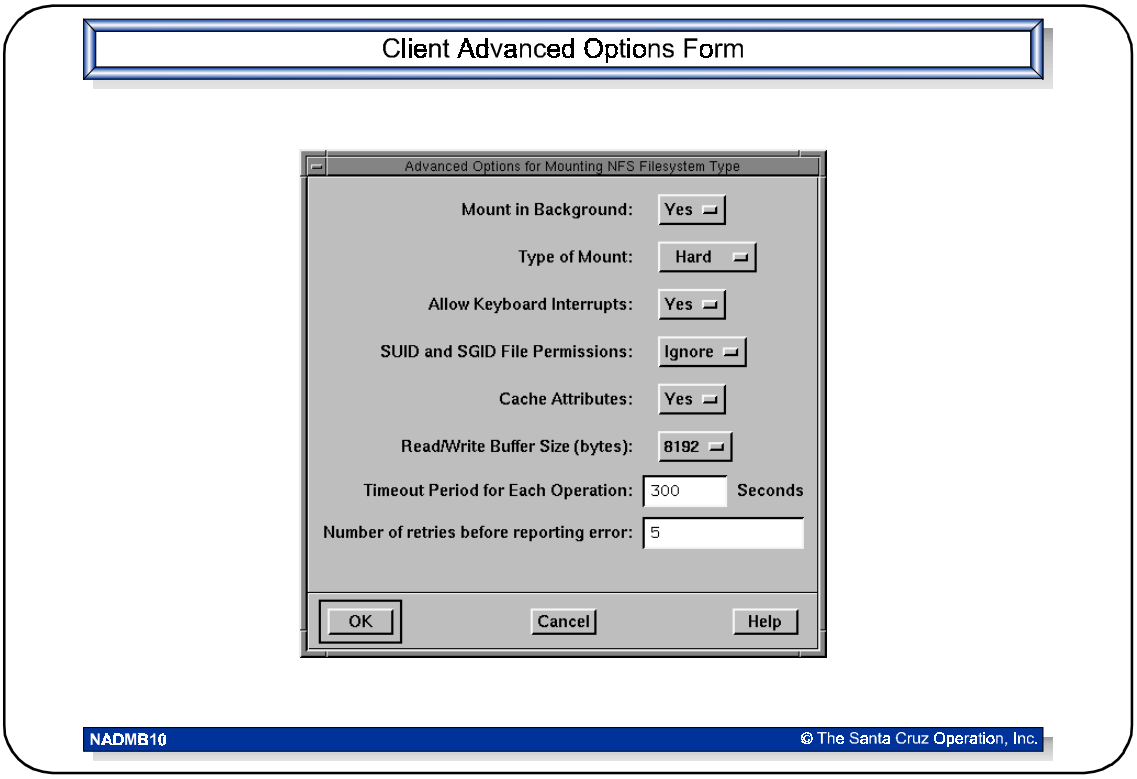
A.2.4 Using scoadmin(ADM)

As with NFS servers, NFS clients can be configured using the **Filesystem Manager**. Select **Filesystems ⇨ Filesystem Manager ⇨ Mount ⇨ Add Mount Configuration ⇨ Remote**.

Fill out the fields shown in AV A-9:

"Remote filesystem type"	Defaults to NFS
"Remote host"	The NFS file server
"Remote directory"	The directory exported by the NFS file server, or a subdirectory thereof
"Mount point"	The local directory on which you want to mount. If this directory does not exist, it is created.
"Description"	An optional string to describe the remote resource
"Access mode"	Select read-only or read-write. The server must export the filesystem read-write for you to use read-write.
"When to Mount"	Specify **Now**, to issue a **mount**(ADM) command when you click on **OK**. If you specify **At System startup**, then **scoadmin**(ADM) makes these entries permanent in the */etc/default/filesys* file.

AV A-10


Client Advanced Options Form
Advanced Options for Mounting NFS Filesystem Type
Mount in Background:
Yes
Type of Mount:
Hard
Allow Keyboard Interrupts:
Yes
SUID and SGID File Permissions:
Ignore
Cache Attributes:
Yes
Read/Write Buffer Size (bytes):
8192
Timeout Period for Each Operation:
300
Seconds
Number of retries before reporting error:
5
OK
Cancel
Help
NADMB10
© The Santa Cruz Operation, Inc.

“Advanced Mount Options”Displays the advanced options popup menu.

The fields in the **Advanced Options** popup menu set additional options to the **mount** command. The options are documented in the **mount**(ADM) manual page. The fields are:

“Mount in Background”	Sets the **bg** option
“Type of Mount”	**soft**—recommended for filesystems whose servers are considered unreliable, or slow **hard**—Used for reliable servers and links **spongy**—like soft, but will not time out during read and write operations
“Allow Keyboard Interrupts”	Sets the **intr** option
“SUID and SGID File Permissions”	Sets (or unsets) the **nosuid** option
“Cache Attributes”	Sets the **noac** option
“Read/Write Buffer Size (bytes)”	Sets the **rsize** and **wsize** options
“Timeout Period for Each Operation”	Sets the **timeo** option
“Retries before reporting error”	Sets the **retrans** option
“Alternate IP port number”	Sets the port number that the **mountd**(ADM) daemon on the server is listening to, if it is not listening on the standard port

A.2.5 Configuring clients by hand

Configuring an NFS client requires that you perform these two steps on the client:

Step 1. Tell the startup scripts which filesystems to remotely mount from which NFS servers, and any other mount options.

- Client **mount** information is kept in the */etc/default/filesys* file.
- The */etc/default/filesys* file is read during the beginning of multiuser mode for local filesystem mounts, and then again during the startup of NFS for NFS filesystem mounts. It can also be used by a manual **mount** command.

Step 2. Mount remote NFS filesystems from the command line after NFS starts up. This is done when you want to mount additional NFS filesystems after the NFS daemons have started.

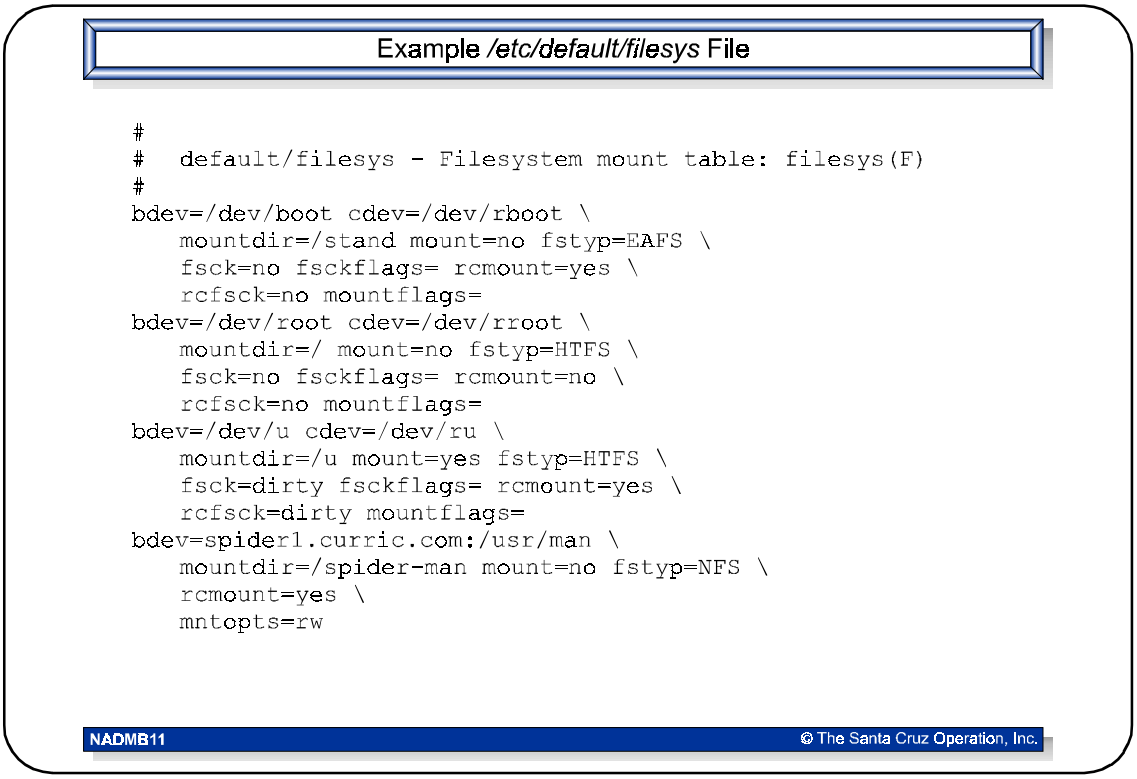
A.2.5.1 The */etc/default/filesys* file

Make entries in the */etc/default/filesys* file to indicate which filesystems on local disks:

- Should be mounted at system startup
- Are available for mounting on demand with the **mount**(ADM) command

Make similar entries in this file to indicate which NFS filesystems should be mounted at system startup. Some of the fields use different syntax than those used for local disks, and some additional fields can be specified.

AV A-11


Example /etc/default/filesys File
#
default/filesys - Filesystem mount table: filesys(F)
#
bdev=/dev/boot cdev=/dev/rboot \
mountdir=/stand mount=no fstyp=EAFS \
fsck=no fsckflags= rcmount=yes \
rcfsck=no mountflags=
bdev=/dev/root cdev=/dev/rroot \
mountdir=/ mount=no fstyp=HTFS \
fsck=no fsckflags= rcmount=no \
rcfsck=no mountflags=
bdev=/dev/u cdev=/dev/ru \
mountdir=/u mount=yes fstyp=HTFS \
fsck=dirty fsckflags= rcmount=yes \
rcfsck=dirty mountflags=
bdev=spider1.curric.com:/usr/man \
mountdir=/spider-man mount=no fstyp=NFS \
rcmount=yes \
mntopts=rw
NADMB11
© The Santa Cruz Operation, Inc.

AV A-11 shows an example */etc/default/filesys* file. The fields in this file are:

`bdev` This field takes the form *`server:directory`* where server is the name of the NFS file server, and directory is the name of the directory on the server to mount.

The server must be preconfigured to allow you to mount the desired directory. NOTE: You can mount any subdirectory of an exported directory; you do not need to mount the entire directory tree.

`fsck` Must be set to `no`; you cannot run **fsck** on an NFS filesystem

`rcfsck` Also set to `no`

`fstype` The type of the filesystem. Set this field to `NFS`

`mntopts` Any options to pass to **mount**(ADM); for example:

```
rw,rsize=1024,wsize=1024
```

Refer to the **mount**(NADM) manual page for additional NFS-specific options. For further information on the */etc/default/filesys* file, see the *filesys*(F) manual page.

A.2.5.2 Manually mounting NFS filesystems

If you need to mount an NFS filesystem temporarily, you can do so:

- If entries are already in the */etc/default/filesys* file, enter:

  `mount` ***`dirname`***

  where *dirname* is the name of the mount point as specified in */etc/default/filesys*.

- If no entry exists in */etc/default/filesys*, use the complete form of the **mount**(NADM) command. The syntax for mounting an NFS filesystem from the command line is:

  `mount -F NFS[,`*`options1`*`] [-o `*`options2`*`] `*`server`*`:`*`/dir /mountdir`

  where:

  options A list of options to **mount**(NADM).

server The name of the NFS file server.

dir The directory exported by the NFS file server, or a subdirectory thereof.

mountdir The directory on the NFS client where the remote directory is attached.

For example:

```
mount -F NFS,rsize=1024,wsize=1024 thatch:/usr/local /usr/local
```

This mounts the */usr/local* directory from *thatch* onto the */usr/local* directory of the current machine. Note that:

- The local and remote directories (*dir* and *mountdir*) do not have to be directly in the root (/) directory.
- You must specify full pathnames to the directories.

The list of NFS-specific options to **mount**(NADM) includes:

soft Normally, mounts are done in hard mode. This means that the client waits indefinitely for each NFS operation. This can cause an NFS client to hang for the duration of an NFS server crash. This option causes mounts to be done in soft mode, which means that both the initial mount and subsequent NFS operations eventually fail if the server does not respond.

spongy Makes the **stat**, **lookup**, **fsstat**, **readlink**, and **readdir** operations follow the semantics of a soft mount, and the rest of the I/O operations to follow the semantics of a hard mount.

bg Normally, mount waits until the NFS server responds, or in the case of a soft mount, it times out. This option causes the **mount** command to perform its work in the background, allowing normal work to proceed. This is an especially good option for the */etc/default/filesys* file, but is less useful on the command line.

rsize and wsize

These two options set the size of the NFS I/O operations. The default:

- Is 8192 bytes
- Can be too large for certain older networking cards which have limited on-board cache (for example, the 3Com 3C501 and Western Digital 8003)

- If too large, it can cause excessive retransmissions because the server drops partial or complete NFS packets.

Mounts made to servers with such cards should adjust the I/O size to 1024.

timeo	Sets the NFS timeout in tenths of a second. This is the time used for soft I/O operations.
retrans	Sets the number of retries before giving up on an I/O operation.
port	Sets the port number.
noac	Turns off caching of file attributes. This allows greater synchronization with the NFS server, for which you will pay a drastic performance penalty.
intr	Allows keyboard interrupts on hard mounts
tcp	Uses the TCP protocol instead of the UDP protocol when communicating with the NFS server

NOTE: Filesystems mounted with the **mount** command from the command line are not automatically remounted when the system reboots. That functionality is provided by the */etc/default/filesys* file.

A.3 NFS Startup and Shutdown

NFS is implemented using many user-level daemons, a small amount of extra code in the kernel, and a networking utility called Remote Procedure Calls (RPC). These are started from the */etc/nfs* script.

A.3.1 The */etc/nfs* script

NFS is started by the */etc/nfs* script, which:

- Has links from the files:
  - */etc/rc0.d/K79nfs*
  - */etc/rc2.d/S89nfs*
- Is run during transitions to the indicated run states using the **start** and **stop** options, as appropriate
- Can be run from the command line with the appropriate argument to start and stop NFS, when needed

When you run this script with the **start** option, it starts NFS by:

- Removing all NFS entries currently in the */etc/exports* file
- Sharing all of the NFS filesystems using the command:

  ```
  exportfs -a
  ```

- Starting the NFS daemons (see Section A.3.1, “The /etc/nfs script,” on page A-28)
- Mounting any NFS filesystems with the command:

  ```
  numountall
  ```

- Starting the automounter (see Section A.5, “The NFS Automounter,” on page A-32)

When you run this script with the **stop** option, it stops NFS by:

- Unmounting all mounted NFS filesystems
- Killing each of the NFS daemons
- Unexporting all of the NFS filesystems

A.4 NFS Security Issues

The use of NFS opens your system up to additional security risks, all of which are manageable, but must be allowed. The following sections discuss each of these issues and how they are addressed.

A.4.1 Setuid and setgid bits on files on NFS servers

Files marked with the *setuid* and *setgid* bits pose a risk to remote systems mounting the files using NFS just as they do the local system on which they physically reside. Further, an NFS file server can have a *setuid* file that, as a network administrator, you do not want to be *setuid* on NFS clients.

NFS solves this problem by allowing you to specify the **nosuid** option to the **mount**(NADM) command. Specifying this flag means that NFS will not honor the *setuid* permission on files accessed through NFS.

A.4.2 Root privileges

All NFS operations pass the UID and GID of the process requesting I/O operations from the client to the server. However, this can be dangerous if the UID is 0, that is, the process on the client is running as **root**. Therefore, NFS normally passes the UID -2 to the NFS server if the UID on the client is **root**.

You can force NFS to pass UID 0 to the server by specifying:

`root=`*`host`*`[:`*`host`*`…]`

where host is a fully qualified host name in the */etc/exports* file, or the corresponding form in the **Filesystem Manager**.

A.4.3 User equivalence revisited

Recall that file access is granted during the **open**(S) system call on UNIX machines. The procedure for determining whether access is granted is:

Step 1. Compare the UID on the process with the UID on the file. If these match, check the first set of permissions and if the requested access is not allowed, stop.

Step 2. If the initial comparison in the previous step fails, compare the GID on the process with the GID on the file. If these match, check the second set of permissions and if the requested access is not allowed, stop.

Step 3. If the comparison in the previous step fails, determine if the user represented by the UID of the process is a member of the group represented by the GID of the file. If they are, use the second set of permissions and if the requested access is not allowed, stop.

Step 4. If all of the above comparisons fail, use the last set of permissions and if the requested access is not allowed, stop.

Notice that the entire process is based upon UIDs and GIDs, not user names or group names. What if:

- A user on an NFS client had the same UID as a different user on the NFS server?
- The GID of the **sales** group on the server was different than that on the client?

In these cases, either more or less access than is intended is provided.

Recall from the discussion on user equivalence in Section 7.3.4, "Overview of user equivalence," on page 7-39 that:

- Enforcing user equivalence prevents these kinds of security problems.
- Keeping user equivalence consistent across a large number of machines on your enterprise network can be a huge amount of work.

The Network Information System (NIS) is a solution to this complexity. See **scohelp ⇨ Networking Documentation Set ⇨ Networking Guide ⇨ Configuring the Network Information Service (NIS)**.

Another good reference is:

Stern, Hal, ***NFS and NIS***, Sebastopol, CA: O'Reilly & Associates, 1992, ISBN: 0--937175-75-7

A.5 The NFS Automounter

Mounting NFS filesystems uses additional resources on both NFS clients and servers, even if I/O operations are not being performed. If your NFS clients only need access to the server's filesystems on an occasional basis, leaving them mounted all the time wastes resources.

The automounter allows you to configure your NFS clients to be ready to mount and unmount remote directories on an as-needed basis. Whenever a user on a client running the automounter invokes a command that needs to access a remote file, the shared resource to which that file belongs is mounted automatically. When a certain amount of time has elapsed, without the resource being accessed, it is automatically unmounted.

A.5.1 The automount process

The automounter operates as follows:

1. The automounter is given a list of directories to monitor, using files called maps, where NFS filesystems need to be mounted when the directory is accessed.

2. When an application attempts to access one of the monitored directories, the **automount**(NADM) daemon performs the NFS mount, but mounts the remote resource to a parallel directory under the directory */tmp_mnt*. For example, if the directory being monitored is */u/user1*, then the automount daemon mounts the remote resource to */tmp_mnt/u/user1*.

3. The automounter creates a symbolic link from the monitored directory to the mount point. Continuing with the example, a symbolic link is created from */u/user1* to */tmp_mnt/u/user1*.

4. Access occurs to the location seamlessly.

5. If the remote resource goes unused for a configurable amount of time, then the automounter unmounts it to save system resources.

Good examples for using the automounter include:

Home directories — If a machine has accounts for several people, many of whom have their home directories on different file servers, then their home directories can be mounted on demand when they log in.

Distributed tools	If various tools are loaded onto servers around the network, and these tools are only used occasionally, you might want to configure the filesystems on which these tools exist to be automounted.
Redundant servers	When a directory is replicated on multiple file servers for redundancy, you can configure the automounter to look at multiple servers when it attempts to mount filesystems.

For additional examples and criteria see the **automount**(NADM) manual page.

NOTE: Automounting is strictly an issue for NFS *clients*. You do not need to do any configuration on your NFS servers beyond that covered in Section A.2, “Configuring Servers,” on page A-11.

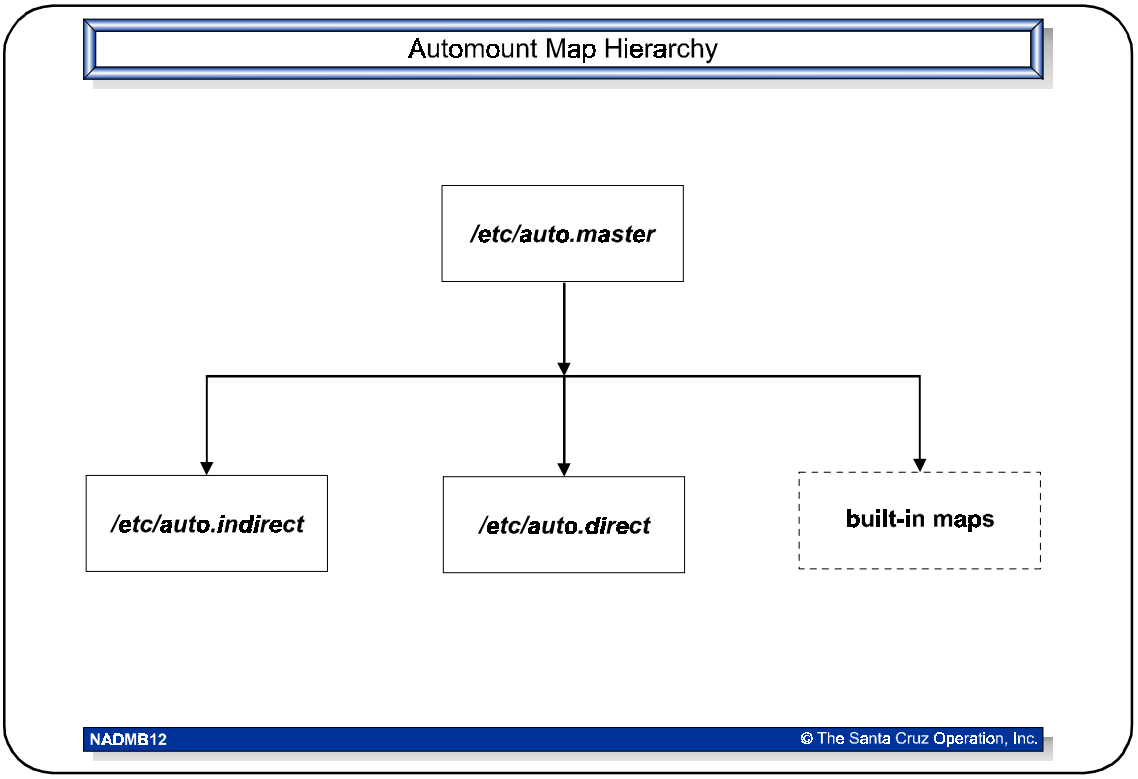
To configure automounting on an NFS client:

1. Ensure that the remote NFS file server is configured to allow mounting of the desired directory.

2. Create at least one automount map on the NFS client. Maps are covered in the next section.

3. Start automounting by issuing the **automount** command, either on the command line or through an entry in the */etc/nfs* startup script.

AV A-12


Automount Map Hierarchy
/etc/auto.master
/etc/auto.indirect
/etc/auto.direct
built-in maps
NADMB12
© The Santa Cruz Operation, Inc.

A.5.2 Configuring maps

Maps tell **automount** which directories are under its control. Although you can configure all directories in a single map, your job can be much simpler if you use a hierarchy of maps. This means that the master map points to other maps. The examples below indicate why this is better.

There are four types of maps in a hierarchy:

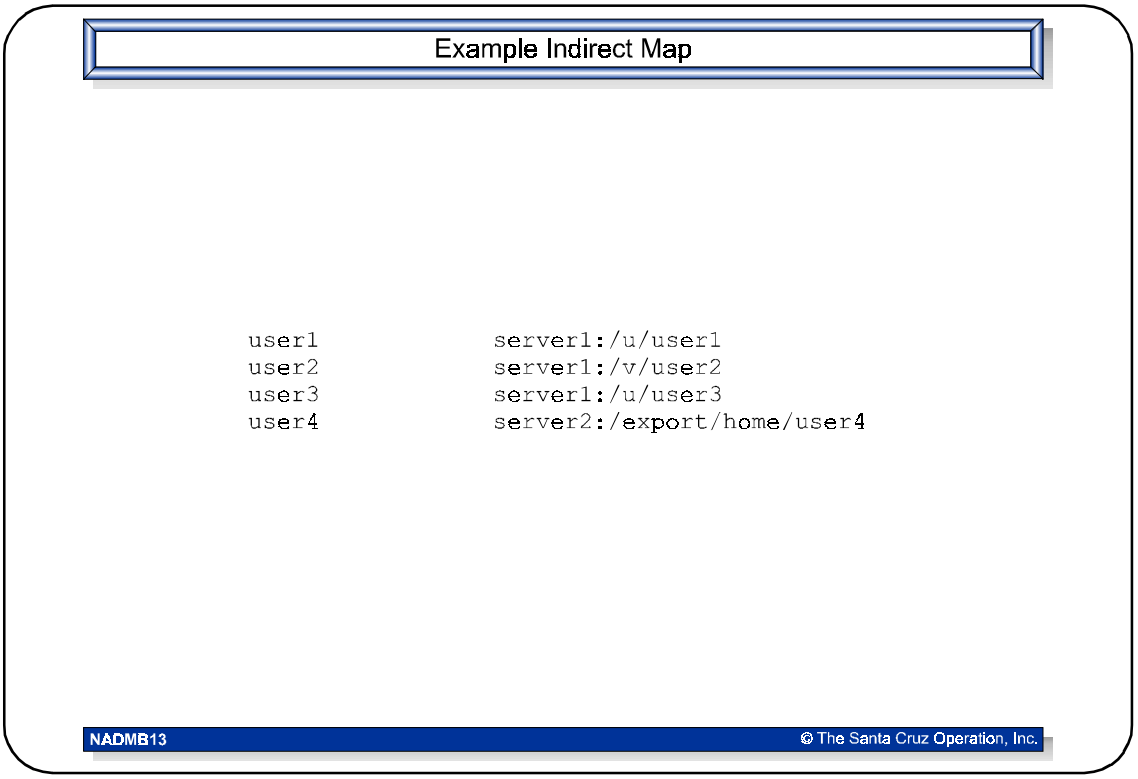
- Master
- Direct
- Indirect
- Built-in

The first field in the master map indicates whether the entry refers to a direct or indirect map.

AV A-12 shows the hierarchical relationship of the maps.

While this might seem overwhelming at first, remember that you have the option of putting everything into one map, and in general, most administrators only use two maps, the direct and indirect map.

AV A-13


Example Indirect Map
user1 server1:/u/user1
user2 server1:/v/user2
user3 server1:/u/user3
user4 server2:/export/home/user4
NADMB13
© The Santa Cruz Operation, Inc.

A.5.2.1 Indirect maps

Indirect maps are the most common kinds of maps because they use the real power of automounting. With indirect maps, the automounter is given:

- An absolute pathname of a directory to monitor and the name of the indirect map associated with that directory in the master map file
- A list of subdirectories that can be NFS mounted under that directory in an indirect map file.

The format of entries in indirect maps is:

subdirectory [*mount_options*] *location* [*location* …]

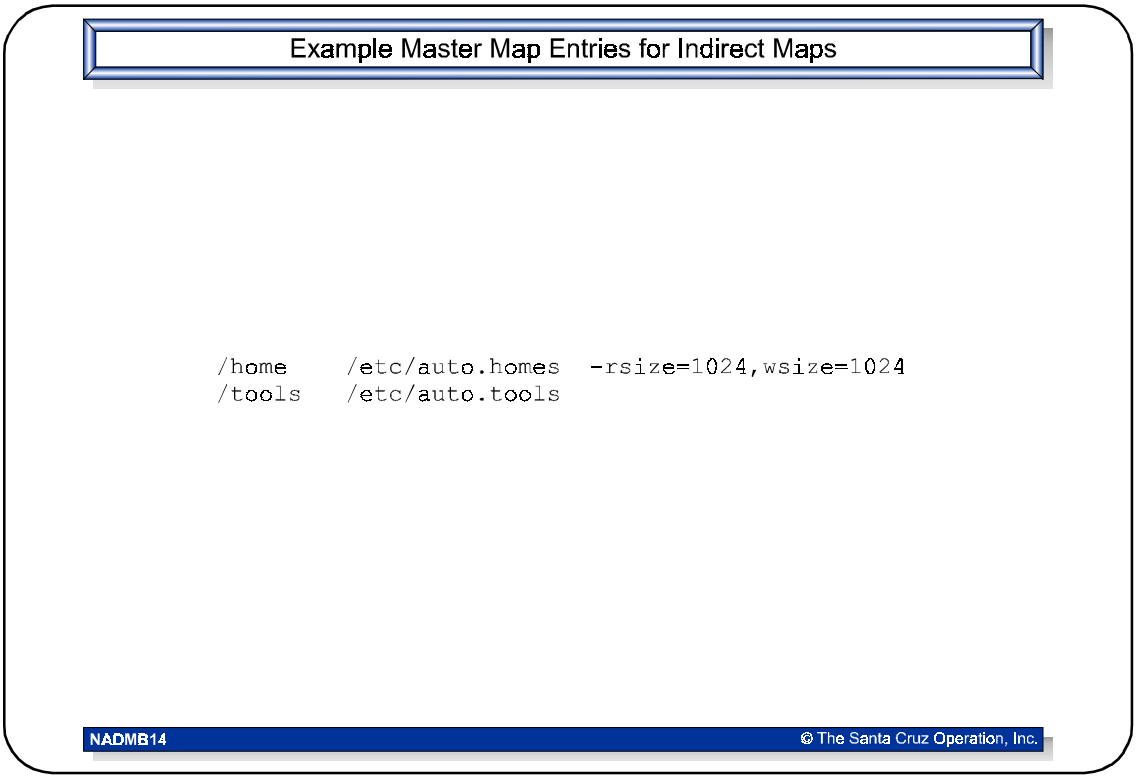
where:

subdirectory	Is the name of the subdirectory under the directory name specified in the master map that should be automounted. For example, if the master listed the */u* directory, this field could be set to `user1`.
mount_options	Are options that you would specify to the **mount**(NADM) command, for example, -ro,bg
location	Is the name of the remote resource, for example, `server1:/homedirs/u1`

Another advantage of the automounter is that you can use it to create an organized file tree out of an otherwise chaotic list of source directories. For example, you can cause all of the home directories on clients to appear under the */home* directory, regardless of where the home directories are on the servers.

In the example in AV A-13, the subdirectory fields list user names: **user1**, **user2**, **user3**, and **user4**. Suppose the direct map that points to this example map references the */home* directory. Therefore, there are four automount directories being controlled: */home/user1*, */home/user2*, */home/user3*, and */home/user4*. This map would typically be stored in the */etc/auto.homes* file. NOTE: You can have multiple indirect maps, in which case you will probably want to name them something descriptive, rather than something like */etc/auto.indirect*.

AV A-14


Example Master Map Entries for Indirect Maps
/home /etc/auto.homes -rsize=1024,wsize=1024
/tools /etc/auto.tools
NADMB14
© The Santa Cruz Operation, Inc.

A.5.2.2 The master map

Remember, the master map tells the **automount**(NADM) daemon which directories to monitor, and name of the file that has the list of other directories associated with that entry. The format of the master map is:

mount_point map_name [*mount_options*]

If you are setting up an indirect map, set the following:

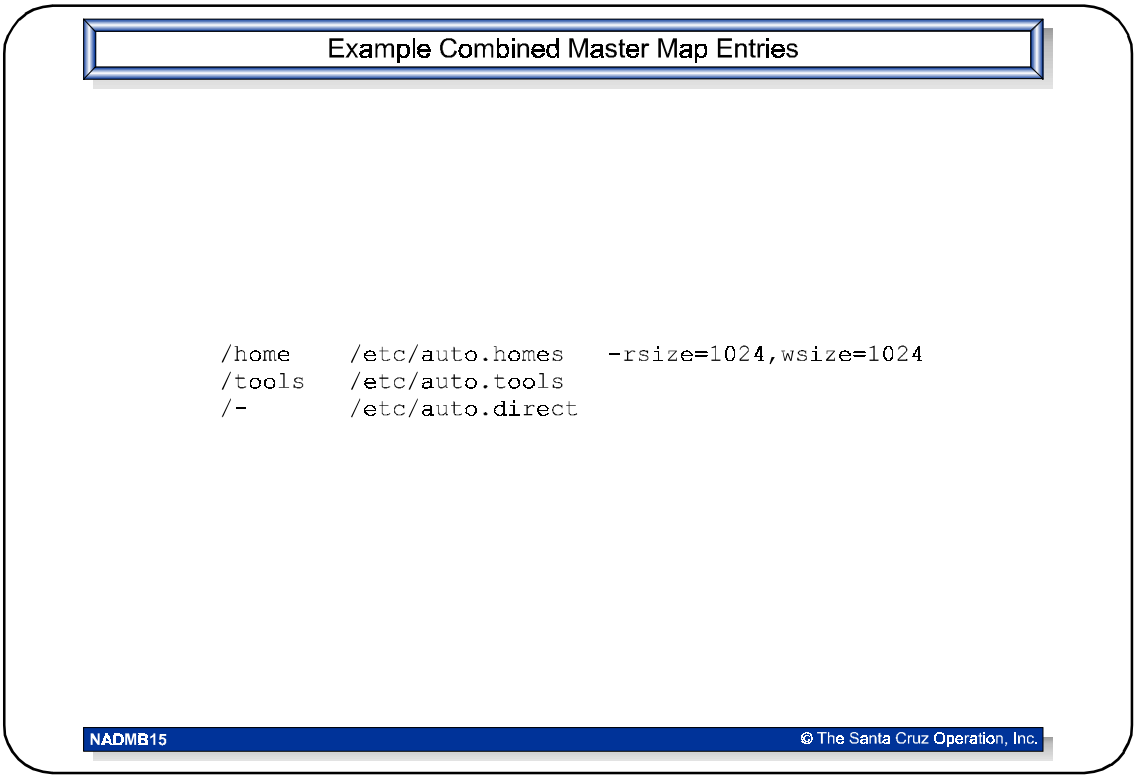
mount_point	The full path name of the directory to be monitored. Keeping with the previous example, this would be set to */home*.
map_name	The file containing the indirect map. For our example, the indirect file was called */etc/auto.homes*.
mount_options	Options you would specify to **mount**(NADM), with the exception of **bg** and **fg**. Any mount options in the indirect map take precedence over these options.

An example master map is shown in AV A-14. This slide shows the use of two indirect maps: *auto.homes* and *auto.tools*.

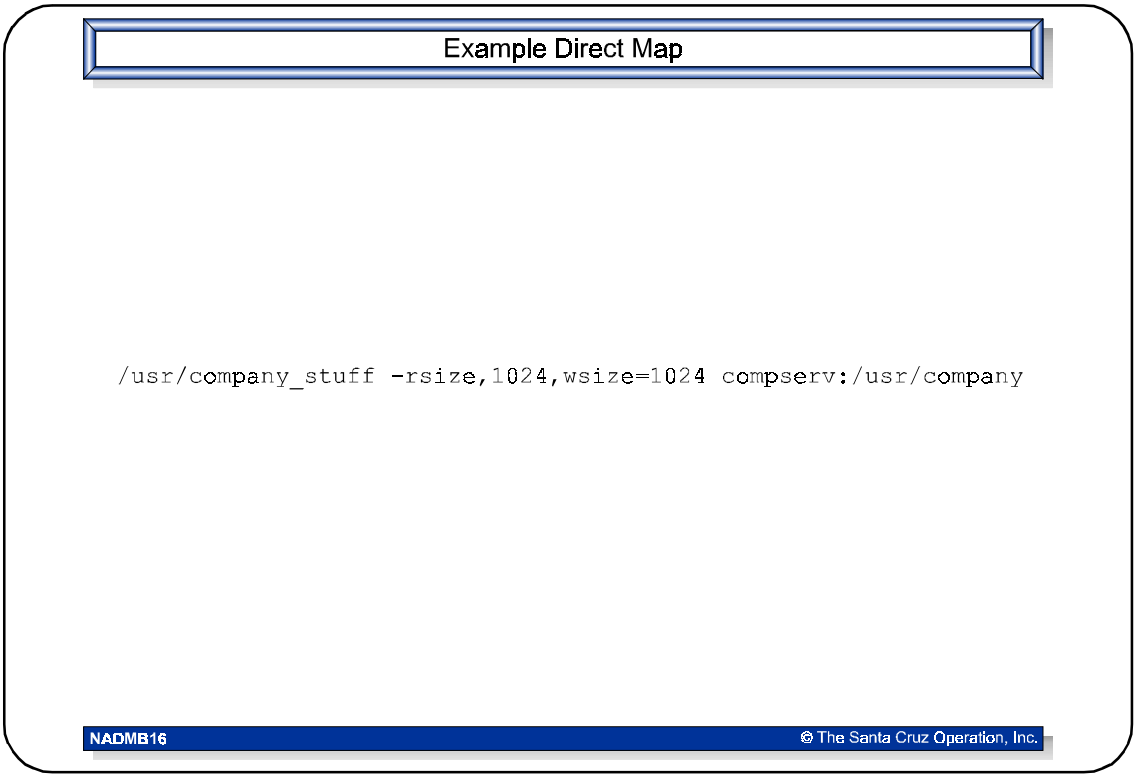
You can also create entries in the master map that point to direct maps. Direct maps are used when you want to always mount the same absolute pathname from some server. For example, suppose you wanted to mount */usr/company_stuff* from an NFS server. This would be a good use of a direct map. In this case, set the following:

mount_point	The two characters `/-` tell the **automount** daemon that this entry is for a direct map. In this case, the absolute pathname of the mount point is found in the direct map file. NOTE: You usually list one direct map entry, unless you want to specify different mount options. This is because a single direct map file can list all the absolute pathnames you want automounted.

AV A-15


Example Combined Master Map Entries
/home /etc/auto.homes -rsize=1024,wsize=1024
/tools /etc/auto.tools
/- /etc/auto.direct
NADMB15
© The Santa Cruz Operation, Inc.

AV A-16


Example Direct Map
/usr/company_stuff -rsize,1024,wsize=1024 compserv:/usr/company
NADMB16
© The Santa Cruz Operation, Inc.

map_name	The file containing the direct map, for example, */etc/auto.direct*.
mount_options	To any of the options you can specify for a **mount**(NADM) command of an NFS filesystem, with the exception of **bg** and **fg**. Any mount options in the direct map take precedence over these options.

AV A-15 shows the same example master map previously displayed, plus an entry for the example direct map.

A.5.2.3 Direct maps

Direct maps have the same format as indirect syntax as indirect maps, that is:

```
dirname [ mount_options ] location [ location … ]
```

There is one important distinction. In indirect maps, the *dirname* names a subdirectory under a directory named in the master map. In direct maps, *dirname* must be an absolute pathname to the directory you want automounted. All other fields are the same as indirect maps described in Section A.5.2.1 on page A-37.

In the previous section, you were presented with an example usage of a direct map, that is, mounting */usr/company_stuff* from an NFS server. AV A-16 shows the entry in the direct map that would cause the mount to happen from a NFS server named compserv.

A.5.3 Configuring the automounter

The **automount**(NADM) daemon is the beginning of all automounting activity. You can start it from the command line. If the */etc/auto.master* file exists, automount is started automatically by the NFS startup script.

The syntax of the automount command is:

```
automount [-mnTv] [ -D envar=value ] [ -f master_file ]
[ -M mount_directory ] [ -t sub-options ]
[ mount_point map [ -mount_options ] ] …
```

All of the flags are described in the **automount**(NADM) manual page. For the purposes of configuration, several are worthy of note.

-f *masterfile*	Indicates which file contains the list of directories for which **automount** is responsible. If you do not specify this argument, **automount** assumes the name of the master map is */etc/auto.master.*

-tl *seconds* Specifies the time in seconds that **automount** will keep an idle filesystem mounted. The default is 300 seconds.

A.6 Lesson Summary

This lesson discussed NFS administration including how to create NFS servers and clients, as well as NFS performance and security issues.

Also presented was the use of the NFS automounter to mount directories on-demand using maps. The following four types of maps were discussed:

- Master maps
- Direct maps
- Indirect maps
- Built-in maps

Review questions

1. Briefly describe what NFS does.

2. What does the */etc/nfs* script do?

3. Which file must be modified on an NFS client to cause an NFS filesystem to be mounted automatically whenever the system goes into multiuser mode?

4. Which file must be modified on an NFS server to cause an NFS filesystem to be exported automatically whenever the system goes into multiuser mode?

5. Normally, NFS does not send UID 0 from client to server. With what does NFS replace this number? How do you configure the server to stop this from happening? Why is this default action a good idea?

6. Why might you choose to use the automounter over the convention mount techniques?

7. Which file does **automount**(NADM) read by default?

A.7 Lab: NFS Administration

The objective of these labs is to give you hands-on experience in configuring NFS clients and servers, configuring and using the automounter, and observing NFS security issues.

Overview

In this lab, you will configure an NFS server and NFS client, adding various options and observing what happens when an NFS file server crashes.

You are to configure two systems in your lab to test NFS.

Procedure

You will need two machines to perform this lab, so you will work with a partner. Decide which of you will be the server and which the client. After you succeed in performing the lab one way, you will do it again reversing roles. You should set up and test the server before you set up the client.

Step 1. Select a directory on the server to export to the client. For example, export */home* or */usr* on the server. The client should mount this directory to */mnt*.

Step 2. Configure the */etc/exports* file on the server.

Step 3. Verify that the directory is exported with **showmount**(NADM).

Step 4. On the client, try and mount the directory exported by the server to your */mnt* directory. If this fails, determine the problem and try again until the mount succeeds.

Step 5. After the mount works from the command line, unmount it.

Step 6. On the client, make an entry in the */etc/default/filesys* file, either with a text editor or using the **Filesystem Manager**. Do not use any advanced options.

Step 7. Reboot the system. From multiuser mode, verify that the filesystem is mounted. If it is not, determine the problem and try again until the mount succeeds.

Step 8. Shut down the file server. Reboot the client with the server still down. What happens?

Step 9. Bring the server back up. What happens to the client?

Step 10. On the client, modify the entry in */etc/default/filesys* to make a soft mount in the foreground. Repeat Steps 10 and 11. How do the results change?

A.8 Lab: Configuring the automounter

The goal of this lab is to use the automounter to mount filesystems.

Overview

You have been a network administrator for several months.

Your organization is growing and you are certain that there is no way you are going to be able to keep up with making sure all those filesystems stay mounted at the right time in the right place.

Further, performance is starting to degrade because so many filesystems are being mounted using NFS that the output of a **df**(ADM) command fills an entire screen!

You will use the automounter to resolve your problems.

Procedure

You can work on your own in this lab. However, we will build on the work accomplished in the previous lab. All you need to do is configure the automounter to mount:

- The filesystem you had previously mounted from */etc/default/filesys*
- User's home directories from the instructor's */home* directory where your student accounts will now have their home directories

Step 1. Remove the entry in your */etc/default/filesys* file that mounted your previous lab partner's directory to your */mnt*.

Step 2. Create an */etc/auto.master* file with two entries in it: a direct entry for */mnt* pointing to the map *auto.mnt*; and an indirect entry for */home* pointing to the map auto.home.

Step 3. Create the *auto.mnt* directory so that it mounts your former lab partner's directory onto your */mnt* directory. This mount should be read-only.

Step 4. Create the *auto.home* map with at least three lines for the users in the */etc/passwd* file that have */home* as their directory: **user1**, **user2**, and **user3**. Do this using wildcards if you can.

Step 5. Restart NFS.

Step 6. Run **df**. Note what filesystems are mounted.

Step 7. Run **cd** to */mnt*. Run **df** again. Note which filesystems are mounted.

Step 8. Run **cd** to */home/user1*, then **cd** to */home/user2*. Run **df** again. Note which filesystems are mounted.

A.9 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

automount(NADM)	Is a daemon that automatically and transparently mounts an NFS filesystem as needed
df(ADM)	Prints the allocation portions of the generic superblock for mounted or unmounted file systems, directories or mounted resources
exportfs(NADM)	Re-exports filesystems on NFS servers
nfsd(NADM)	Performs NFS I/O on NFS servers
mount(ADM)	Mounts local and remote NFS resources
mountd(NADM)	Mounts remote NFS resources
numountall(NADM)	Unmounts multiple filesystems
showmount(NADM)	Shows all remote mounts

Relevant files

/etc/auto.home	Default direct or indirect map filename. This filename is included by default in the NIS makefile.
/etc/auto.master	Is the default master map file for the NFS automounter
/etc/default/filesys	Contains a list of filesystems to mount on NFS clients
/etc/exports	Contains a list of filesystems to export on NFS servers
/etc/nfs	Is the NFS startup script
/etc/xtab	Contains a list of filesystems currently exported
/tmp_mnt	Is the parent directory for NFS automounter mounts

A.10 Answers to Review Questions

This section provides suggested answers to review questions.

1. The Network File System (NFS) allows transparent access of files and directories located on a NFS server.

2. The */etc/nfs* script starts and stops the NFS daemons.

3. The */etc/default/filesys* file must be modified on NFS clients to cause an NFS filesystem to be mounted automatically whenever the system goes into multiuser mode.

4. The */etc/exports* file must be modified on an NFS server to cause an NFS filesystem to be exported automatically whenever the system goes into multiuser mode.

5. NFS changes the UID 0 to UID -2. To stop this from happening, you can set the `root=hostlist` option in */etc/exports*. This default action is a good idea because **root** on a client cannot be trusted as **root** on the file server.

6. The automounter might use less resources than a full time mount for filesystems that are accessed infrequently.

7. The **automount**(NADM) daemon, reads the master map file */etc/auto.master*.

B

Network Boot and Remote Configuration

Instructor Guide

Appendix B

Network Boot and Remote Configuration

Suggested appendix duration

This appendix should take approximately 1 hour to complete.

Pre-appendix classroom preparation

Have devices that boot off the network readily available for lab and demonstration. Such devices can include:

- X-terminals
- Network printers
- Terminal servers

Instructor preparation

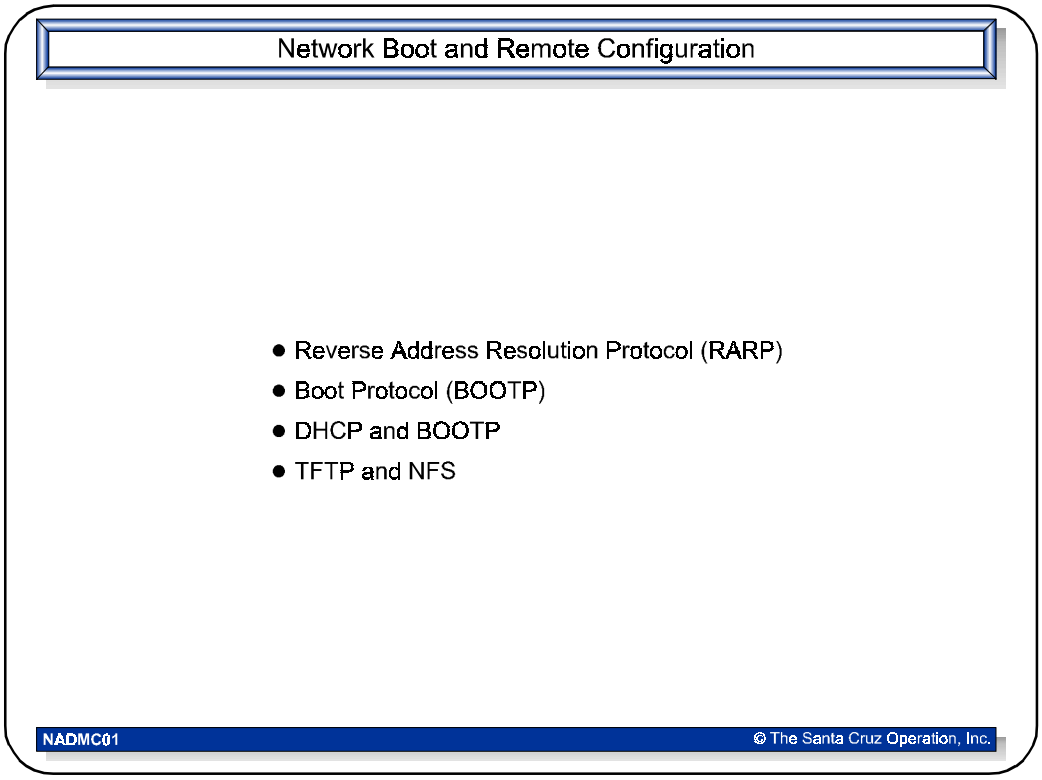
You may want to review the relationship between **bootpd**, **dhcpd**, **rarpd,** and **aasd**.

The RFCs are a good source of information about DHCP, BOOTP, and their interactions. Specifically:

- 1531 covers DHCP.
- 1533 covers the DHCP parameters.
- 1534 covers the interaction between DHCP and BOOTP.

A good web site for DHCP information is *http://www.keil.co.kr/~nova/bookmark/bookmark.html*.

AV B-1


Network Boot and Remote Configuration
Reverse Address Resolution Protocol (RARP)
Boot Protocol (BOOTP)
DHCP and BOOTP
TFTP and NFS
NADMC01
© The Santa Cruz Operation, Inc.

Introduction

Lesson 11: “Windows Connectivity” introduced the use of DHCP to provide remote boot configuration to Windows systems. However, DHCP is not the only such protocol available. Many other devices on the network, such as printers, X-terminals, terminal servers, and routers, can be configured from a SCO OpenServer Release 5 machine using one of several protocols.

This appendix shows you how to configure the boot protocol (BOOTP) and reverse ARP (RARP) daemons to boot and configure various devices remotely. It also covers the relationship between these protocols and DHCP.

Objectives

By the end of this appendix, you will be able to:

- Describe what the RARP protocol is, what it is used for, and how to configure it
- Describe what the boot protocol (BOOTP) is, what it is used for, and how to configure it
- Describe the relationship between the DHCP and BOOTP
- Describe two alternatives for downloading a device’s kernel

AV B-1: Network Boot and Remote Configuration
This slide describes the objectives covered in this appendix.

B.1 Overview of Booting from the Network

You might encounter devices that connect to your network and need to download their kernels and/or their IP information (address, routers, DNS servers, etc.) from a server on the network. Examples of these devices include:

- Network printers
- Terminal servers
- X terminals
- Diskless workstations
- Network Computers (NCs)
- Some Windows clients
- Computers that are being installed over the network

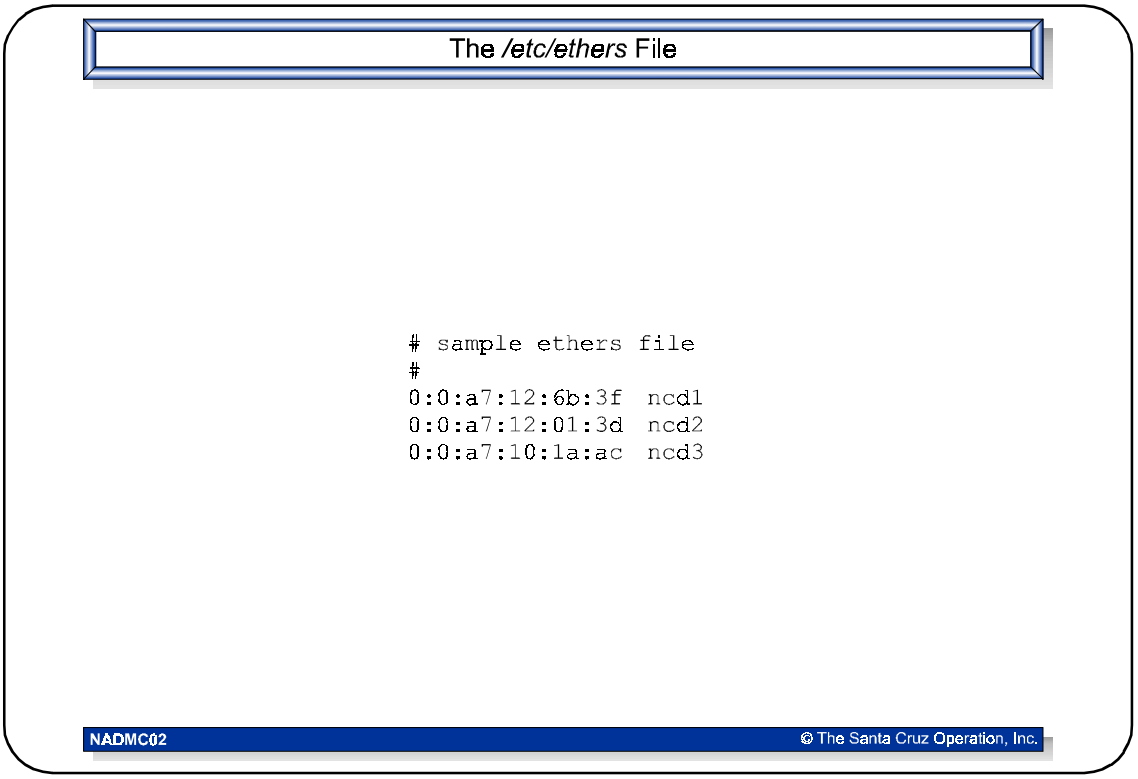
These devices need to do one or more of the following:

- Obtain their IP information if it is not stored in some non-volatile RAM (NVRAM) or flash PROMS
- Find a download server, unless the name of the server is stored in NVRAM or flash PROMS
- Initiate the download

There are three ways that devices booting from the network can get their boot information:

- The reverse address resolution protocol (RARP) allows devices to discover their IP address based upon their Ethernet addresses.
- BOOTP allows devices to discover other things, including the IP address of name servers, their netmask, and more.
- DHCP is an extension to BOOTP that allows additional information to be sent to the client as well as a dynamic IP address.

AV B-2


The /etc/ethers File
sample ethers file
#
0:0:a7:12:6b:3f ncd1
0:0:a7:12:01:3d ncd2
0:0:a7:10:1a:ac ncd3
NADMC02
© The Santa Cruz Operation, Inc.

B.2 The RARP method, rarpd(ADMN), and the */etc/ethers* file

Recall that if a device has a host's IP address and needs its corresponding Ethernet address, then:

- An ARP message is broadcast on the local network.
- The host with the corresponding IP address responds with its Ethernet address.

RARP:

- Is a reverse-ARP protocol
- Is used by a device to find out its IP address, given its hardware address
- Is only usable on Ethernet networks
- Uses the **rarpd**(ADMN) daemon to answer RARP requests

Usually only one machine on a local network is running **rarpd**. When **rarpd** hears a RARP-REQUEST being broadcast on its local network, it:

1. Looks up the Ethernet address of the requestor in its */etc/ethers* file.
2. Sends back the corresponding IP address.

AV B-2: The */etc/ethers* File
Use this slide to demonstrate the format and usage of the */etc/ethers* file.

AV B-2 illustrates the format of the */etc/ethers* file along with a sample entry. The */etc/ethers* file contains only two fields:

- The Ethernet address of a device
- The IP address or a host name that can be resolved to an IP address, for that device

You can also find this information in the *ethers*(SFF) manual page.

The **rarpd**(ADMN) daemon can be run once for each interface on a machine. For example, if a machine:

- Has two interfaces called net0 and net1
- The two interfaces have IP addresses 200.100.50.1 and 200.100.51.1 respectively

The **rarpd** daemon can be started with the following two lines:

```
/etc/rarpd net0 200.100.50.1

/etc/rarpd net1 200.100.51.1
```

The **rarpd**(ADMN) daemon can also be started for all interfaces using this command line:

```
/etc/rarpd -a
```

If */etc/ethers* exists, **rarpd** is started by default during TCP startup.

RARP only provides IP addresses. Additional information is provided by either:

- BOOTP
- DHCP

Each of these are discussed in the next sections.

B.3 The BOOTP Protocol

The BOOTP protocol allows devices booting from the network to get information other than the IP address from a BOOTP server. To allow devices on your network to use BOOTP, you must configure a BOOTP server on each network segment containing a BOOTP client. This server runs the boot protocol daemon, **bootpd**(ADMN).

The following sequence of events occurs for devices using BOOTP, if the device and server are on the same subnet:

1. The device broadcasts a BOOTREQUEST packet.

2. The **bootpd** daemon on each configured server looks in their database file, typically the */etc/bootptab* file, to determine whether the device with the given Ethernet address is known.

3. If the entry for the device is found, **bootpd** sends the device a BOOTREPLY packet containing all of the information in that device's entry in the */etc/bootptab* file.

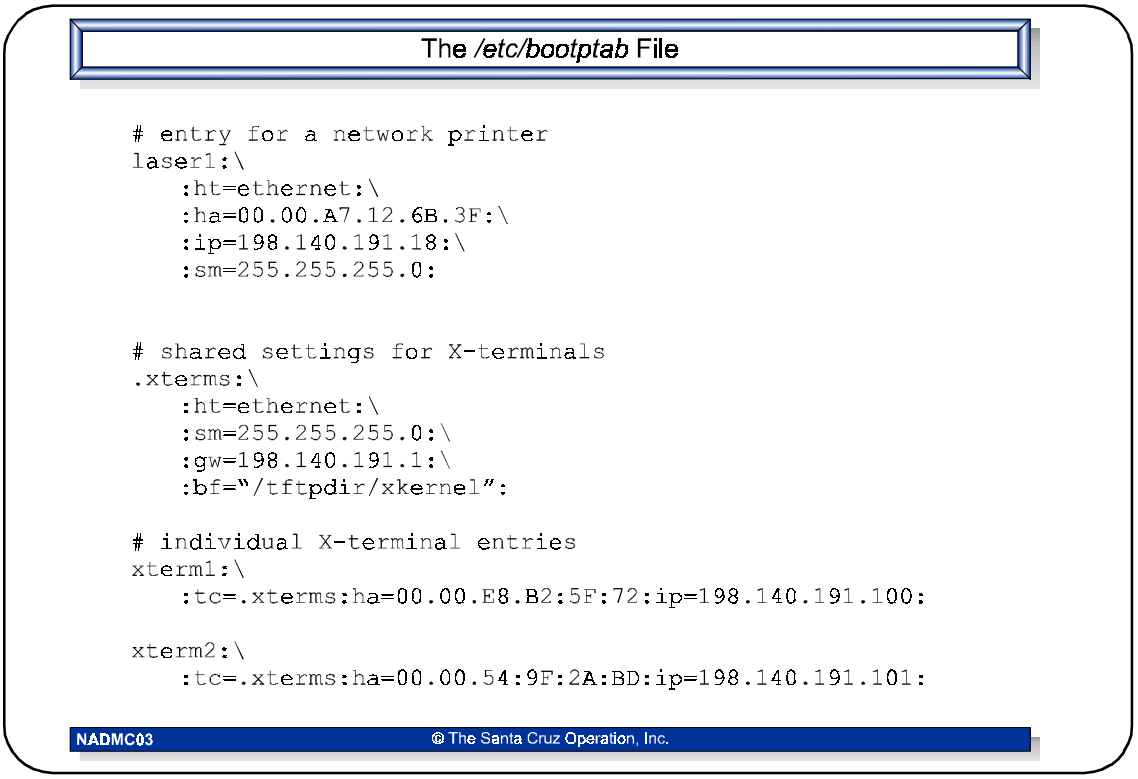
Discuss the kinds of information typically found in a *bootptab*(SFF) file.

The information you can put into the *bootptab*(SFF) file includes:

- IP address
- Netmask
- Name of file that holds the kernel for this device
- Name of server that holds the kernel
- IP address of domain name server
- Domain name

The **bootpd** daemon does not actually transfer the kernel to the client. That must be accomplished using one of the transfer methods discussed later in this lesson.

AV B-3


The /etc/bootptab File
entry for a network printer
laser1:\
:ht=ethernet:\
:ha=00.00.A7.12.6B.3F:\
:ip=198.140.191.18:\
:sm=255.255.255.0:
shared settings for X-terminals
.xterms:\
:ht=ethernet:\
:sm=255.255.255.0:\
:gw=198.140.191.1:\
:bf="/tftpdir/xkernel":
individual X-terminal entries
xterm1:\
:tc=.xterms:ha=00.00.E8.B2:5F:72:ip=198.140.191.100:
xterm2:\
:tc=.xterms:ha=00.00.54:9F:2A:BD:ip=198.140.191.101:
NADMC03
© The Santa Cruz Operation, Inc.

Should you have these devices on your network, your job as a network administrator is to:

- Configure the **bootpd**(ADMN) to run as described in Section B.3.2, “Starting the bootpd daemon,” on page B-14.

- Configure the */etc/bootptab* file to contain one entry for each device on a network that requires this service.

NOTE: Because BOOTREQUEST packets are sent out with broadcast packets, they are heard only on the local network. If subnets or multiple networks are in use, then you must also configure **bootpgw**(ADMN), the boot protocol gateway, to run from **inetd**(ADMN) on the routers.

B.3.1 The */etc/bootptab* file

AV B-3: The */etc/bootptab* file.
Use this slide to illustrate the entries possible in a *bootptab* file.

AV B-3 shows sample *bootptab* entries. Information about the parameters is available in the *bootptab*(SFF) manual page.

You can configure your devices to get many of these pieces of information from other locations. BOOTP allows you to keep all of this related information together in a database. If your device uses BOOTP, you should put as much information into *bootptab* entries as you can.

Examples of distributing this data is to have the kernel of a diskless workstation tell the workstation its name, or to have the PROMS of an X terminal know the IP address of the name servers.

Exercise: Creating a *bootptab* entry

On paper, create an entry you would put into the *bootptab* file for an Ethernet terminal server with the following characteristics:

- The Ethernet address is 00:00:C0:FA:27:83.

- The IP address is 200.100.50.75.

- Its bootfile is */tftpboot/tskern3.0.2*.

B.3.2 Starting the bootpd daemon

There is no BOOTP manager that can be used to configure the */etc/bootptab* file or the **bootpd**(ADMN) daemon. To start the **bootpd** daemon at system boot, uncomment the line provided for it in the */etc/inetd.conf* file so that the line looks like this:

```
bootps dgram udp wait root /etc/bootpd bootpd
```

If you are going to use **bootpd** and DHCP, you must run **bootpd** in "slave mode" using the **-S** option. In this case, uncomment these lines in */etc/inetd.conf*:

```
bootps dgram/i udp wait root  /etc/dhcpd  dhcpd -b bootps-alt

bootps-alt dgram udp wait root/etc/bootpdbootpd -S:
```

After you have made the appropriate changes to the */etc/inetd.conf* file, the **DHCP Server Manager** will use heuristics to make future changes consistent with your previous changes.

For example, if you had previously uncommented the **bootpd**(ADMN) entry in the */etc/inetd.conf* file and then used the **DHCP Server Manager** to configure **dhcpd**(ADMN), the **DHCP Server Manager** would make the changes to */etc/inetd.conf* that allow **bootpd** and **dhcpd** to coexist.

Instructor Guide

There are problems running **bootpd** and DHCP services together because they both use the same UDP port 67. See Section B.3.2, "Starting the bootpd daemon," on page B-14 for more information.

B.4 The DHCP Protocol

BOOTP is meant for relatively static environments where:

- Devices are not added or removed from the network routinely.
- Every device is always attached to the same subnet and uses the same IP address.

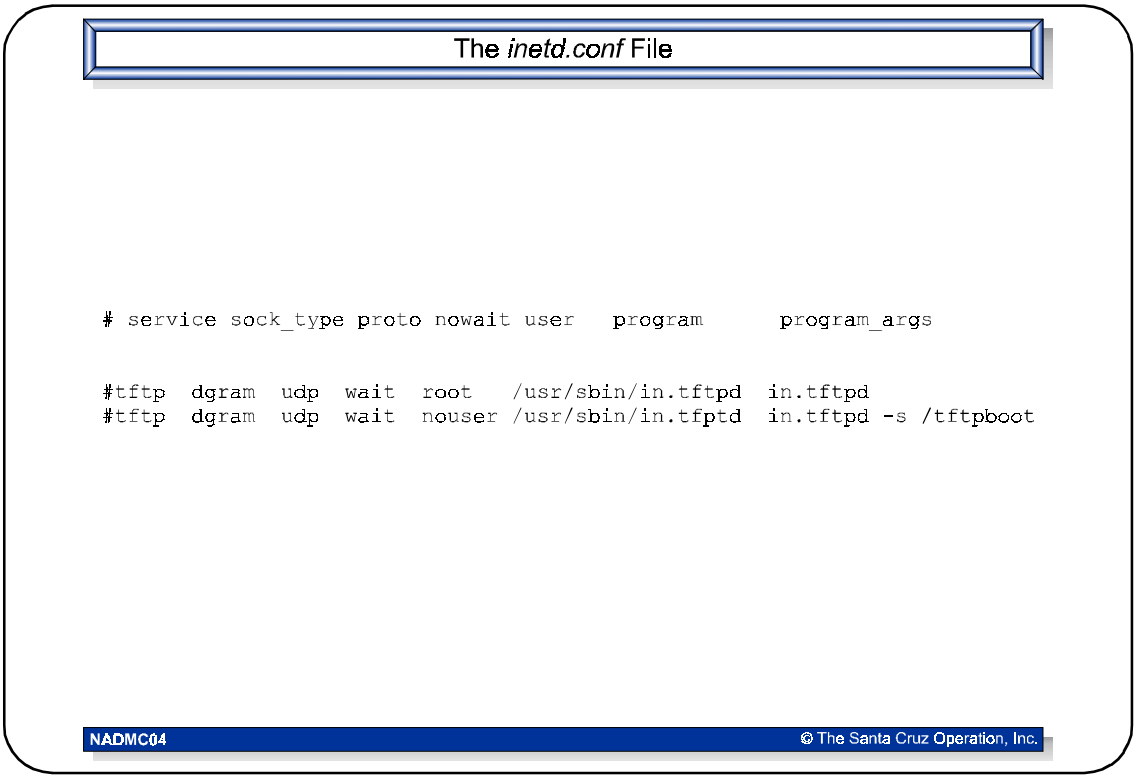
Your environment might not match this paradigm. Problems you can encounter include:

- Devices that roam your intranet and need different IP addresses depending upon where they are attached. For example:
  - Support people who take their notebooks or other network devices to a user's area and need network access while working in that area
  - Auditors and other personnel that travel to various geographically distinct locations in the company that need network access
- Visitors to your company that you allow to access your company network
- Training rooms with network connections available to students bringing their own notebooks

In all of these examples, computers need to obtain a new IP address, and possibly a new name, when they get plugged into the network.

The Dynamic Host Configuration Protocol (DHCP) is an extension to BOOTP that solves this and other problems. DHCP was introduced in Section 11.4.2, "Configuring the DHCP server," on page 11-57.

AV B-4


The inetd.conf File
service sock_type proto nowait user program program_args
#tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd
#tftp dgram udp wait nouser /usr/sbin/in.tfptd in.tftpd -s /tftpboot
NADMC04
© The Santa Cruz Operation, Inc.

B.5 Downloading the Device's Kernel

After a device that boots from the network knows its IP address (and possibly other information if it uses BOOTP), it can download its kernel. There are two methods commonly used to download kernels, TFTP and NFS.

B.5.1 The TFTP transfer method

AV B-4: The *inetd.conf* File
Use this slide to point out the two commented versions of **tftpd** in */etc/inetd.conf*.

One way a device can get its kernel is to transfer it using the Trivial File Transfer Protocol (TFTP). This protocol is a variant of FTP that does not authenticate users. This presents a significant potential security risk.

Most devices use TFTP. As a network administrator with such a device on your network, you must:

- Modify the */etc/inetd.conf* file to enable the **tftpd**(ADMN) daemon, and restart **inetd**(ADMN). Which of the two **tftpd** entries you should choose depends upon your security concerns. See Section B.5.3, "Security concerns," on page B-17.

- Place the kernel on the machine running **tftpd**.

Additional information about TFTP can be found in the **tftp**(TC) and **tftpd**(ADMN) manual pages.

B.5.2 The NFS transfer method

Some devices use the Network File System (NFS®) as an alternative to TFTP. NFS file servers can list which machines they allow to mount their filesystems across the network, on a host-by-host and filesystem-by-filesystem basis.

The actions you must take to set up an NFS file server are discussed in Appendix A: "NFS".

B.5.3 Security concerns

You should be aware of several security concerns when using DHCP, BOOTP, RARP, and TFTP. The following are a set of general rules you should follow when using these protocols:

Request on local networks only	Never have a gateway broadcast or rebroadcast (with **bootpgw**(ADMN)) any of these packets to networks outside your own enterprise network, particularly, onto the Internet.

Discuss why this is so. For an example, use the concept of a LAN device that needs to download a kernel with passwords broadcasting a BOOTP message and a node outside the enterprise replying that it has the appropriate kernel *and password*. This outside node could then take control of the device!

Only accept local requests	Never allow TFTP packets from outside your enterprise network into your network.

Discuss why this is so. For an example, use the concept of an external machine trying to download files from your machines, such as updated versions of kernels. NOTE: This and the previous security tip both require packet filtering knowledge, which is beyond the scope of this course.

Restrict TFTP directory access	Start **tftpd**(ADMN) with the option **`-s`** *`dir`*, where *dir* is the name of a directory that contains only download files. This prevents malicious people from downloading any publicly readable, yet network-sensitive files, such as */etc/passwd*.

Instructor Guide

Discuss why */etc/passwd* should be guarded from view by users outside the system.

B.6 Lesson Summary

This lesson has described how to boot devices from the network and assign them static and dynamic IP addresses.

Review questions

1. What is the name of the server daemon that answers RARP requests and what file does it get its data from?

2. What is the name of the server daemon that answers BOOTP requests and what file does it get its data from?

3. Name one problem present in BOOTP that is solved by DHCP.

B.7 Relevant Commands and Files

The following commands and files were presented in this lesson.

Relevant commands

bootpd(ADMN)	The boot protocol server daemon
bootpgw(ADMN)	The boot protocol gateway server daemon
dhcpd(ADMN)	The dynamic host control protocol daemon
inetd(ADMN)	The Internet super daemon
rarpd(ADMN)	The reverse ARP server daemon
tftpd(ADMN)	The trivial file transfer protocol server daemon

Relevant files

/etc/ethers	Contains translations from Ethernet to IP addresses
/etc/bootptab	Contains BOOTP configuration information
/etc/inetd.conf	Is the configuration for **inetd**(ADMN)
/tftpboot	Is the directory typically used to hold secure files available via **tftp**

2.8 Answers to Review Questions

This section provides suggested answers to review questions.

1. The **rarpd**(ADMN) daemon answers RARP requests and gets its information from the */etc/ethers* file. See Section B.2, “The RARP method, rarpd(ADMN), and the /etc/ethers file,” on page B-9.

2. The **bootpd**(ADMN) daemon answers BOOTP requests and gets its information from the */etc/bootptab* file. See Section B.3, “The BOOTP Protocol,” on page B-11 and Section B.3.1, “The /etc/bootptab file,” on page B-13 respectively.

3. DHCP solves the need to dynamically allocate IP addresses to a network device. See Section B.3, “The BOOTP Protocol,” on page B-11.