SCO® UNIX® Operating System

System Administrator's Guide

SCO®

OPEN SYSTEMS SOFTWARE

SCO® UNIX®
Operating System
System Administrator's
Guide

SCO®
OPEN SYSTEMS SOFTWARE

Date: 31 January 1992
Document Version: 3.2.4C

Chapter 1 Introduction

The UNIX system is designed to accommodate a wide variety of needs, including the use of peripheral devices, including terminals, printers, and tape drives. Your UNIX system also requires careful control of its operation and a regular schedule of maintenance. This guide explains how to run and maintain the operating system on your computer, ensuring maximum performance with the fewest problems.

The tasks presented in this guide range from simple ones requiring very little knowledge about UNIX systems, to complex tasks requiring extensive knowledge about the operating system and your computer. Each chapter explains the tools and knowledge you need to complete the tasks described in that chapter. In some cases, you may be referred to other manuals.

This guide contains chapters about computer hardware you may wish to use with your system. The use and interaction of various devices with the operating system is described in a comprehensive fashion. For example, "Using floppy disks and tape drives" discusses the use of magnetic storage media, and it covers the basics of preparing the operating system for such a device, installing it, and how to use the drive once it is installed.

This guide also explains how to expand your system with remote communications over phone lines; UUCP can be set up to communicate with UNIX sites all over the world. (See "Building a remote network with UUCP" in this guide for a complete explanation of network facilities available.) An important part of system operation is the protection of data on the system. Security is discussed in great detail in "Maintaining system security." The system includes flexible mechanisms designed to protect your data.

Pay special attention to "Troubleshooting your system." This chapter is an excellent resource to help you keep your system running smoothly. Refer to this chapter whenever you run into difficulties.

Documentation conventions

The following documentation conventions are used in this guide.

boldface	Commands are shown in **boldface**. For example:
	... the **fsck** command examines the various structures on the disk and ...
	UNIX system utilities or library routines are also shown in **boldface**. For example:
	... the installation program runs the **fdisk**(ADM) utility ...
	Notice in this example that the location of the corresponding manual page is also given. A full list of manual page locations is given in Table 1.2.
	Literal user input is also shown in **boldface**. For example:
	... to display the file itself, enter: **more /etc/termcap** and press ⟨Return⟩ ...
italics	Directories and filenames are shown in *italics*. For example:
	... the set of administrative print service commands is contained in the */usr/lib* directory ...
	Emphasized words or phrases are also shown in *italics*. For example:
	... the constant creation and removal of files creates a situation called *disk fragmentation* ...
	References to book titles are also shown in *italics*, but with initial capitals. For example:
	... for information relating to system use, refer to the *User's Guide* ...

bold italics	Placeholders are shown in ***bold italics***. A placeholder is a word which you must replace with an appropriate filename, number, or option. For example: ... do you want to use the ***mouse_type*** on any other terminals ... In this example, ***mouse_type*** would be replaced by a brand or type of mouse.

`courier`	Screen displays and other output from the computer are shown in `courier`. For example:
	 `The UNIX kernel has been rebuilt.`
	 `Do you want this kernel to boot by default? (y/n)`

`reverse video`	When you are using **sysadmsh**, the context indicator, status line, and cursor position are shown in `reverse video`. See chapter 2 of this manual for examples.
" "	Data values and field names are shown in "quotation" marks. For example: ... where *x* is "0" for a display adapter or "1" for a serial port ... Quotation marks are also used for normal words used in a way particular to computing. For example: ... the former is called the "master" tty and the latter is called the "slave" tty ... Document chapter names are also shown in quotations. For example: ... consult the "Administering serial terminals" chapter of this guide ...
SMALL CAPITALS	Acronyms are shown in SMALL CAPITALS. For example: ... the name UUCP is an acronym for UNIX to UNIX Copy ...

SMALL BOLD CAPITALS

System parameters (definable system values, for example, the number of disk drives attached to the system), named constants (programming names with fixed values) and environment variables (definable system information, for example, what type of terminal is being used) are shown in **SMALL BOLD CAPITALS**. For example:

... the **mkdev pty** updates the **NSPTTYS** kernel parameter ...

... the preferable method for setting your terminal type is to assign the type to the **TERM** variable ...

⟨ ⟩

Names of keys are shown in ⟨angle brackets⟩. For example:

... press the ⟨Esc⟩ key to exit the current mode ...

Δ

Where **sysadmsh** menu selections are given as alternatives to command line entries they are indicated by Δ . For example:

... to copy all the files in the directory */u/bogart* to the cartridge drive */dev/rct0*, enter:

```
tar cvf /dev/rct0 /u/bogart
```

Δ **sysadmsh** users select: Media ⇨ Archive ...

The system administrator and administrative roles

Every UNIX system should have at least one person in charge of system maintenance and operation. In this guide, such a person is called a system administrator. It is the responsibility of system administrators to ensure the smooth operation of the system and to perform a wide variety of tasks that require special privileges.

You can choose to have a single system administrator or divide the tasks among several persons, each charged with a different area of operation. You can even assign roles that are strictly limited to one aspect of the system.

Depending on the size of the system and the number of users on it, system administration can be anything from a once-a-day task to a full-time job. Even if the system is small, the system administrator should consistently perform each required maintenance task, because sloppy maintenance can adversely affect system performance.

The system administrator should keep a hard copy log of all system modifications and system events. Each event, message, backup, or modification should be logged with the date, time, and name of the person logging, and the circumstances surrounding the event. For example, if a new application is added to the system software, an entry should be placed in the log. This entry should include the time, date, and name of the person installing, and any notes about the software or installation that may be helpful. An accurate log helps in diagnosing system problems and charting the growth and use of a system.

All tasks in this guide are presented from a system administrator's point of view, but many can also be accomplished by ordinary users. Because some of the tasks dramatically change the system's operation, we recommend that, whenever possible, the system administrator perform these tasks. However, no matter who performs an operation, it should be entered in the system log. Following these rules can prevent unwanted or unnecessary changes to the system.

A system administrator has several tasks to perform, sometimes on a daily basis:

- Make certain the integrity of the system is not compromised through use of security mechanisms.
- Make certain that adequate backups (regular copies of files on the system) are made and stored for future use.
- Handle problems related to use of limited computer resources (disk space, number of processes, and so on).
- Alleviate system communication (network) stoppages due to failed connections.
- Apply operating system updates and maintenance fixes.
- Provide general support to users.

Summary of administrator's tasks

A system administrator has numerous tasks to perform. They can be divided into groups according to how often they are carried out. The following list of tasks ranges from those that must be performed more often than once a day to those that need be performed less often than once a month. The administrator may have to do some of the tasks in the following list more or less often, depending upon the size and complexity of the system. For more information on each task, see the reference indicated.

Table 1-1 Task list

As Needed Tasks	For More Information:
☐ Record all system modifications and events in log.	Preceding paragraphs in this section
☐ Be on call for panics, crashes, power spikes, user questions.	Chapter 20, "Troubleshooting your system"
☐ Maintain security of hardware, software, data file access.	Chapter 11, "Maintaining system security" and chapters on using each of the hardware components

Daily Tasks	For More Information:
☐ Perform backups.	Chapter 16, "Backing up filesystems"
☐ Check usage levels.	"Using performance tools to diagnose system inefficiency" in Chapter 17, "Tuning system performance"
☐ Check for runaway processes.	"Stopping a runaway process" in Chapter 20, "Troubleshooting your system"
☐ Check disk space.	"Displaying free space" in Chapter 5, "Managing filesystems"
☐ Check mail functionality, connections.	"Maintaining the MMDF system" in Chapter 19, "Setting up electronic mail"
☐ Check printer status.	"Examining a printer configuration" in Chapter 7, "Using printers"
☐ Check auditing output, if activated.	"Report generation" in Chapter 12, "Using the audit subsystem"

(Continued on next page)

(Continued)

Daily Tasks	For More Information:
☐ Check communications links, if active.	Chapter 18, "Building a remote network with UUCP"
☐ Check for unattended login sessions.	Man page for **who**(C) command in the *User's Reference*, and "Activity report generation" in Chapter 11, "Maintaining system security"
☐ Remove *core* and **.out* files.	"Maintaining free space in filesystems" in Chapter 5, "Managing filesystems"

Weekly Tasks	For More Information:
☐ Run **fsck**(ADM) on all filesystems.	"Filesystem integrity" in Chapter 5, "Managing filesystems"
☐ Check printer spooler status report.	Check **lp**(C) account mailbox for messages
☐ Check log files such as */etc/wtmp* and those in */usr/adm* and */usr/spool* and clear, trim, or truncate.	"Maintaining free space in filesystems" in Chapter 5, "Managing filesystems"
☐ Use **sar**(ADM) to generate a report of activity.	"Using performance tools to diagnose system inefficiency" in Chapter 17, "Tuning system performance"
☐ Generate detailed report of user disk utilization.	"Displaying disk usage" in Chapter 5, "Managing filesystems"
☐ Remove temporary files and *lost+found* files.	"Maintaining free space in filesystems" in Chapter 5, "Managing filesystems"

Monthly Tasks	For More Information:
☐ Perform full system backup.	Chapter 16, "Backing up filesystems"
☐ Archive critical files if changed.	"Archiving files on tape" and "Using floppies for file storage" in Chapter 8, "Using floppy disks and tape drives"
☐ Re-tune system and re-allocate resources, if necessary.	Chapter 1, "Introduction" and the "Reallocating kernel resources with configure" section in Chapter 17, "Tuning system performance"
☐ Perform hardware maintenance.	Chapter 8, "Using floppy disks and tape drives" and hardware documentation
☐ Change dial-in passwords, if necessary.	"Adding passwords for dial-in lines" in Chapter 15, "Using modems"
☐ Change root password, if necessary.	"Changing a user password" in Chapter 4, "Administering user accounts"

Occasional Tasks	For More Information:
☐ Upgrade OS and Application software, as needed.	*Installation Notes* of the new version
☐ Fix permissions on software.	**fixperm**(ADM) man page in *System Administrator's Reference*
☐ Re-distribute space in filesystems.	"Checking for free space on filesystems" in Chapter 20, "Troubleshooting your system" or "Maintaining free space in filesystems" in Chapter 5, "Managing filesystems"
☐ Find SUID or SGID files, check owner, size.	"Locating files" in Chapter 5, "Managing filesystems"
☐ Locate huge (over 64 Mbyte) files and verify their purpose.	"Locating files" in Chapter 5, "Managing filesystems"
☐ Find "orphan" files (no real user).	"Locating files" in Chapter 5, "Managing filesystems"
☐ Locate sparse directories and compress if necessary.	"Maintaining efficient filesystem organization" of Chapter 5, "Managing filesystems"

Making administration easier with the sysadmsh

The **sysadmsh**(ADM) command is a menu interface designed to simplify the task of system administration. The menus, submenus, and screens allow you to simply point and pick, or fill in blank fields. The **sysadmsh** allows less-experienced system administrators to use UNIX system commands that would otherwise require memorization and constant referring to manual pages. The **sysadmsh** includes context-sensitive help; simply press the ⟨F1⟩ key from any menu to display further explanations of the menu options.

If you are new to UNIX operating systems, we strongly recommend that you become familiar with the concepts and tasks covered in the *Tutorial*. This guide assumes some familiarity with UNIX systems; after studying the *Tutorial*, you should be able to perform the basic system administrative tasks described here.

To aid users of **sysadmsh**, the documentation of this guide is supplemented by **sysadmsh** references that appear below UNIX system command-line instructions.

For example, the following instructions refer to the **custom** utility, used to add more software to your system. Following the command is a sequence of **sysadmsh** menu selections.

Enter the following command:

```
custom
```

Δ **sysadmsh** users select: System ⇨ Software

This means that you can access the functions of the **custom** command by first selecting System at the main **sysadmsh** menu, followed by selecting Software at the next lower level. Selections can be made from the menu in any of the following ways:

- Move through the menu options using the ⟨Space⟩ key and press ⟨Return⟩ on the option you want.
- Move left and right through the options using the arrow keys and press ⟨Return⟩ on the required option.
- Press the first letter of the option required. This is the quickest way. Using the example above, you would simply enter **ss** (without the ⟨Return⟩ key) to reach the **custom** menu.
- Move through the menu options with a mouse, clicking the left button to select an option.

For more instructions on using the **sysadmsh**, refer to the "sysadmsh: using the system administration shell" chapter in this guide.

Locating manual pages

When you use the command line rather than the **sysadmsh** menu interface, you have direct access to utilities and data. Notice the form used for commands in this guide. Each command is printed in bold type, and each has a suffix to help you find more information about it.

The following table lists the locations of the manual pages for the commands with the indicated suffixes. To find information about a command, note the letter or letters that appear in parentheses following the command, then look up the command in the appropriate reference book or guide. For example, the command **lpstat**(C) is defined in the Commands (C) section of the *User's Reference.*

Using online manual pages

If manual pages are installed on your system they may be viewed by typing:

man ***command***

where ***command*** is the command for which you want to see the manual page.

For example, to see the manual page for the **more** command, type:

man more

Some manual pages appear in more than one location (see the following table). To see all occurrences of a particular manual page, type:

man -a ***command***

For example, to see all the manual pages for the **hd** command, type:

man -a hd

The order (by location) in which manual pages are displayed is determined by the */etc/default/man* file. You may edit this file to alter the display order.

To force the system to display a manual page for a particular location, type:

man ***location command***

For example, to see the manual page for the **hd** command for the location **HW**, type:

man HW hd

Table 1-2 Manual page locations

Command Suffix	Book and Purpose
ADM	*System Administrator's Reference* - commands reserved for the exclusive use of system administrators
C	*User's Reference* - operating-system commands available to all users
CP	*Programmer's Reference* - programming commands used with the development system
DOS	*Programmer's Reference* - DOS routines used with the development system
F	*System Administrator's Reference* - (File Formats) description of system configuration files
FP	*Programmer's Reference* - (File Formats) description of system files and data structures
HW	*System Administrator's Reference* - information about hardware devices and device nodes
K	*Device Driver Writer's Guide* - routines provided in the kernel for writing device drivers
M	*User's Reference* - miscellaneous information used for access to devices, system maintenance, and communications
S	*Programmer's Reference* - system calls and library routines for C and assembly-language programming

NOTE The *Programmer's Reference* and *Device Driver Writer's Guide* are only supplied if the Development System is purchased.

The super user account

The super user login (also known as *root*) is a special account for performing system maintenance tasks. It gives the system administrator unusual privileges that ordinary users do not have, such as accessing all files in the system, and executing privileged commands. Many of the tasks presented in this guide require that the system administrator be logged in as the super user. To do this, the system administrator must know the super user password created during the installation of your system. (See the *Installation Guide.*)

Log in as the super user only to perform system-maintenance tasks. Even if the system administrator is the only user of the system, that person should create a user account for day-to-day work, reserving the super user account for system-maintenance tasks only.

Few users should know the super user password. Misuse of the super user powers by naive users can result in a loss of data, programs, and even the operating system itself.

The keyboard

Many keys and key combinations perform special actions on UNIX systems. These actions have names that may not correspond to the keytop labels on your keyboard. Table 1.3 shows which keys on a typical terminal correspond to special actions on UNIX systems. A list for your particular login device is in **keyboard**(HW). Many of these keys can be modified by the user; see **stty**(C).

Table 1-3 Special keys

UNIX Name	Action
⟨Return⟩	terminates a command line and initiates an action. This key is also called the ⟨Enter⟩ key; the keytop may indicate a down-left arrow.
⟨Esc⟩	exits the current mode; for example, exits insert mode when in the editor **vi**. This is also known as the ⟨ESCAPE⟩ key.
⟨Del⟩	stops the current program, returning to the shell prompt. This key is also known as the INTERRUPT key.
⟨Bksp⟩	deletes the character to the left of the cursor. The keytop may show a large left arrow, as opposed to the small "cursor left" arrow.
⟨Ctrl⟩d	signals the end of input from the keyboard; exits the current shell, or logs you out if the current shell is the login shell. This is not interchangeable with the ⟨Break⟩ key.
⟨Ctrl⟩h	deletes the first character to the left of the cursor. This is also called the ⟨ERASE⟩ key.
⟨Ctrl⟩q	restarts printing after it is stopped with ⟨Ctrl⟩s.
⟨Ctrl⟩s	stops printing at the standard output device, such as a terminal. This keystroke does not stop the program.
⟨Ctrl⟩u	deletes all characters on the current line. This is also called the ⟨KILL⟩ key.
⟨Ctrl⟩\	quits current command, creates a *core* file. This is also called the ⟨QUIT⟩ key. (Use of this keystroke is recommended for debugging only; see **core**(FP).)

Running programs simultaneously with MultiScreen

With MultiScreen™, you can run several programs on your console at the same time. You see the display for each program on a different "screen," but you never have to leave your single console. Pressing a simple key combination switches you from one screen to another, and each screen acts independently from the others.

When you log in normally, you see the screen associated with the ⟨Alt⟩⟨F1⟩ key combination. To open a second screen, press and hold the ⟨Alt⟩ key, then press ⟨F2⟩ or another function key on your keyboard. Function keys are generally located across the top or down the far left side of your keyboard.

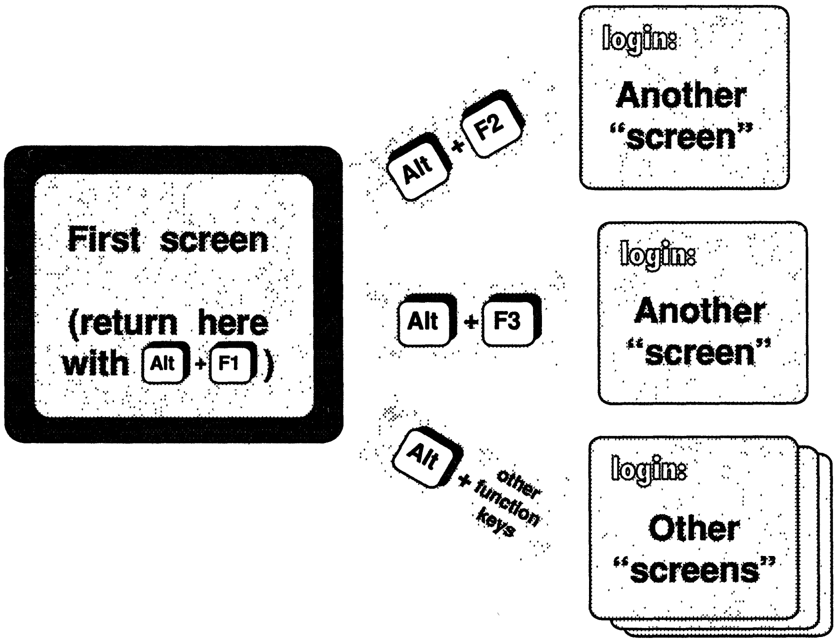


Figure 1-1 MultiScreen example

After you press the key combination, the screen clears and a login prompt appears. Log in again and you can begin work on the second "screen." Press ⟨Alt⟩⟨F1⟩ to switch instantly back to the first screen. Switching between screens in this way can speed up procedures that require working in two different programs. You can run both programs simultaneously, and work on either one at any time. See Figure 1-1.

For example, you can start **sysadmsh** on your first screen, then press ⟨Alt⟩⟨F2⟩ to create a second screen. Login again and begin editing a log file with vi. Use ⟨Alt⟩⟨F1⟩ and ⟨Alt⟩⟨F2⟩ to switch back and forth between screens, taking actions in **sysadmsh** and recording them in the log.

You can open more than two multiscreens at once. However, make sure to log out of all extra screens before logging out of your first screen. Unattended screens where you remain logged in allow unauthorized access to the system.

If you have several screens open at once, you can rotate through them by pressing the Control and Print Screen key combination, ⟨Ctrl⟩⟨PrtSc⟩. This is helpful if you cannot remember which process is running on a particular screen.

Note that you can configure the system to use ⟨Ctrl⟩⟨Alt⟩ ***function-key*** combinations in addition to ⟨Alt⟩ ***function-key*** combinations to change multiscreens. This is especially useful in applications that reserve the ⟨Alt⟩ ***function-key*** combinations for their own use. This can be configured using the **mapkey**(ADM) utility.

For more information, refer to **multiscreen**(M) and **screen**(HW).

System security

An important consideration is protecting the system and its data from unauthorized access. This system includes security mechanisms not found on other UNIX systems. These mechanisms are designed to meet the C2 class of "trust" as defined by the Trusted Computer System Evaluation Criteria (also known as the *Orange Book*). As the system administrator, you can configure the protection mechanisms to the requirements of your site. You can also set up the powerful auditing features to keep detailed records of logins and system usage. The "Administering user accounts" chapter explains how to add users to the system and how to configure the default security scheme. The "Maintaining system security" chapter covers all aspects of trusted operation, and the "Using the audit subsystem" chapter explains how to use the auditing features.

Educating users

The following list contains items the system administrator can explain to users so they can take advantage of the system's resources without overloading them or causing unnecessary system problems. The more users understand the system and its limits, the less demands are placed on the system administrator.

If a user is completely new to UNIX systems, the system administrator should recommend a training course or at least a careful review of the *Tutorial*. Augment this training with the items listed here.

What Every User Should Know:

- System security level - Inform the user whether security is relaxed or not, and if it is not, what files and directories the user can access, and what system actions can be performed. See Chapter 11, "Maintaining system security."
- How to log in - Describe, if permitted, how to generate a user password. Emphasize password secrecy, and methods of memorization. Explain the limits on password length and type, the schedule for changing passwords, rules about changing passwords, the number of tries allowed. Tell users how to specify their terminal type, and help them preset the terminal type in their *.login* or *.profile* files if they use the same type of terminal regularly. See Chapter 4, "Administering user accounts."
- How to manage files and directories - Make sure the user is familiar with basic commands. Refer the user to the *Tutorial* if necessary.
- How to edit files - Make sure the user knows basic **vi** commands, or the basic commands of the default editor. Refer the user to the *Tutorial* or a training class, if necessary.
- How to invoke programs - Relate the names, locations, and commands necessary to run the most-used programs.
- How to print files - Explain appropriate print commands for output of most-used programs. Tell the user the location of the default print service printer. Demonstrate how to replenish paper and toner, tape, or ribbon cartridge. Refer the user to the printer documentation, if necessary.
- Good mail etiquette - Explain how to read and send mail, how to glean through mail lists, and how to organize messages by appending them to mailbox files. Demonstrate how these files can be read with the **mail -f** command. Explain how to clean out unwanted messages by deleting or saving them to floppy. Emphasize the importance of maintaining free disk space.

- Good filesystem planning - Explain the limits of directory size. For best performance, login and working directories should have less than 64* entries (including the dot (.) and dot dot (..) entries), and data storage directories should have less than 638* entries. Warn users that directories do not get smaller, even if entries are removed. Discourage users from saving mail messages in separate files rather than appending them to existing mailbox files. See Chapter 5, "Managing filesystems."
- How to store (archive) files - Show users how to use **tar**(C) or **cpio**(C) to archive unused files or directories to tape or floppy disk. See Chapter 8, "Using floppy disks and tape drives."
- How to reset a scrambled terminal - Teach users how to escape from most-used programs and how to reset a terminal with **tset**(C) and **stty sane**. Tell users how to turn the terminal on and off if necessary. Make sure users try these procedures before asking a system administrator to disable and re-enable the scrambled terminal. See Chapter 20, "Troubleshooting your system."
- How to kill a hung process - Instruct users how to use **ps -flu** on a neighbor's terminal to find the hung process on their own terminal, and how to kill the process without causing undue system problems. See Chapter 20, "Troubleshooting your system."

Site planning considerations

Before installing or expanding your system, review the following list of considerations with management and create a physical plan of the system that makes the most efficient use of available resources, and allows users the most direct and complete access to those resources.

If growth is expected, plan for it. Make sure that the resources and the communications lines that connect them are adequate for expected growth.

- Set up the computers in a place where they will not be bumped or moved at any time. If possible, they should be in a room by themselves, with little or no foot traffic. If workstations are used for data storage, at least place them on stable furniture and leave no cables exposed to traffic.
- Keep the computer room cool and give each machine excellent ventilation; keep all machines away from walls and, if possible, provide a separate air conditioner for the computer room, with more-than-adequate cooling capability.

* These figures apply to filenames of 14 characters or less. As filename lengths increase, up to a maximum of 255 characters, the number of files that fit on a single disk block decreases, thus reducing the optimum number of files in a directory.

- Install a Halon fire extinguishing system in the computer room rather than sprinklers.
- Store backup media in a separate room from the computers. This room should be fireproof, or should at least have a Halon fire extinguishing system rather than a sprinkler system.
- Ensure that there is adequate and uninterrupted power for the computers. This means more than enough current, and at least surge suppressors, if not a means of guaranteeing uninterrupted power supply. If power fluctuations and failures are common, provide a backup power supply. The computers should also be on an isolated, fully grounded (earthed) circuit.
- If you install a Local Area Network, plan the cabling and location of all machines and peripherals carefully. Seek the assistance of a networking expert to make these plans. Good planning and the use of adequate connecting media and compatible hardware are essential for long-term network performance.
- If you need modem lines for off-site connections, arrange for these with your local telephone company.
- If you plan to connect a printer to a parallel port, locate it close to the machine running it, but keep the machine out of the path of traffic to and from the printer.
- If you connect terminals, printers, or other peripherals to serial ports, consider using phone-line cabling and switching hardware, especially if your system is expected to grow in size and complexity. You can readily adapt phone lines for serial hardware, and telephone connecting and switching technology is mature and flexible.

Extensions to the UNIX operating system

A number of features described in this manual represent extensions to the AT&T System V/386 UNIX Release 3.2 base. These features are added value from the Santa Cruz Operation, Inc., and are summarized in Table 1.4. Features that are part of the AT&T System V UNIX Release 3.2 base, but include added value, are marked with a dagger "†".

Table 1-4 Value-Added features

addxusers(ADM)	dbmbuild(ADM)	man(C)	sd(ADM)
ale(ADM)	default(F)	mapkey(ADM)	screen(HW)
ap(ADM)	deliver(ADM)	mcconfig(F)	setcolor(C)
asroot(ADM)	dos(C)	menumerge(ADM)	sg(C)
audit(HW)	dtox(C)	mkdev(ADM)	shutdown(ADM)†
auditd(ADM)	eisa(ADM)	mkfs(ADM)†	su(C)†
authcap(F)	fdisk(C)†	mmdfalias(ADM)	submit(ADM)
authck(ADM)	filesys(F)	mmdftailor(F)	subsystem(M)
authckrc(ADM)	fixmog(ADM)	mnlist(ADM)	sysadmsh(ADM)
autoboot(ADM)†	fixperm(ADM)†	mscreen(M)	tables(F)
badtrk(ADM)	fsave(ADM)	multiscreen(M)	tail(C)
boot(HW)†	fsck(ADM)†	passwd(C)†	tape(C)
cdrom(HW)	fsphoto(ADM)	prwarn(C)	tcbck(ADM)
checkaddr(ADM)	goodpw(ADM)	queue(F)	ttyupd(ADM)
checkmail(C)	hwconfig(C)	rcp(C)	uname(C)†
checkque(ADM)	idleout(M)	rcvalert(C)	unretire(ADM)
checkup(ADM)	integrity(ADM)	rcvfile(C)	usemouse(C)
cleanque(ADM)	kbmode(ADM)	rcvprint(C)	uuinstall(ADM)
cleantmp(ADM)	link_unix(ADM)	rcvtrip(C)	uulist(ADM)
cnvtmbox(ADM)	ln(C)†	relax(ADM)	vidi(C)
configure(ADM)	lock(C)	rmuser(ADM)	xbackup(ADM)
cps(ADM)	logs(F)	sar(ADM)†	xdumpdir(ADM)
custom(ADM)†	maildelivery(F)	schedule(ADM)	xrestore(ADM)

In addition, certain chapters describe functionality that is entirely added value. These chapters are listed in Table 1.5.

Table 1-5 Value-Added chapters

No.	Name	Description
2	"sysadmsh: Using the system administration shell"	This chapter describes the **sysadmsh**(ADM) menu system, which covers all aspects of system administration.
4	"Administering user accounts"	The Accounts branch of the **sysadmsh** is described in this chapter, in addition to extensions to the **passwd**(C) command.
9	"Adding mice and other graphic input devices"	The functionality of **mkdev mouse** and the **usemouse**(C) utility are described here.
11	"Maintaining system security"	This chapter describes the security features designed to meet the C2 level of trust, which are a feature of this version of the UNIX system.
12	"Using the audit subsystem"	This chapter describes the audit subsystem that creates security-related log records. This is part of the security features discussed in Chapter 11.
16	"Backing up filesystems"	The Backups branch of the **sysadmsh** is described in this chapter, including the **schedule**(ADM) file that controls scheduled backups.
19	"Setting up electronic mail"	An enhanced version of the MMDF mailer is described here.
C	"Using the system console and color displays"	The value-added utilities **kbmode**(ADM), **multiscreen**(M), **vidi**(C), and **setcolor**(C) are described in this chapter.

Chapter 2

sysadmsh: using the system administration shell

The **sysadmsh** (**sys**tem **adm**inistration **sh**ell) is a menu interface designed to simplify the task of system administration. The **sysadmsh** allows you to run the numerous system administration commands with their various options without having to use the traditional UNIX system command line.

This chapter explains how to use the **sysadmsh** interface. To use **sysadmsh** effectively, you also need to know something about the UNIX system commands called by **sysadmsh**. Where appropriate, command line equivalents to **sysadmsh** menu options are included in the text. However, you should note that some **sysadmsh** options don't have command line equivalents, for example, none of the **audit** functions do.

You will find it easier to learn the material in this chapter if you start the **sysadmsh** and actually run the examples as you get to them.

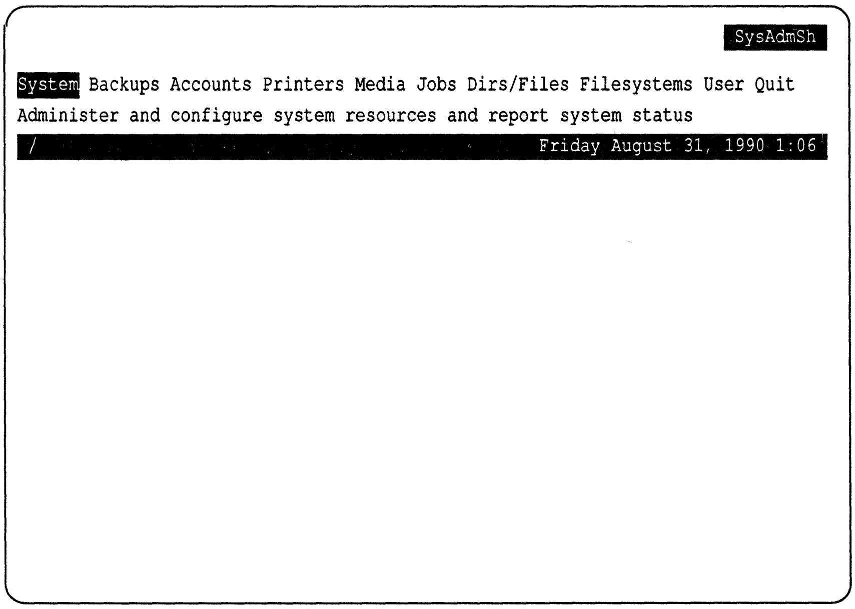
This chapter assumes that you have some knowledge of the UNIX operating system. You should become familiar with the concepts covered in the *Tutorial* before using the **sysadmsh** options.

Starting sysadmsh

To gain access to all the functionality of **sysadmsh,** log in as *root* and enter the following command:

sysadmsh

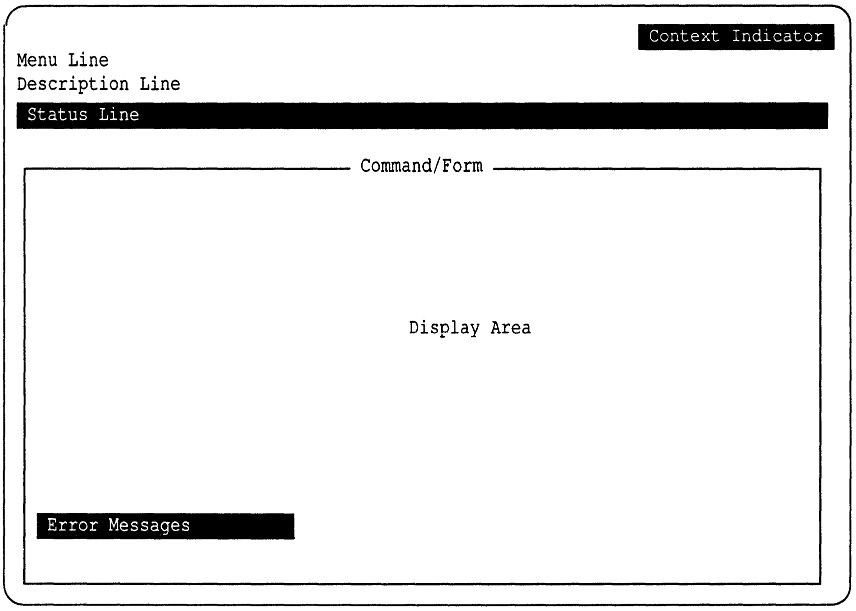
The main **sysadmsh** menu is displayed:



```
                                                                  SysAdmSh

System Backups Accounts Printers Media Jobs Dirs/Files Filesystems User Quit
Administer and configure system resources and report system status
 /                                              Friday August 31, 1990 1:06
```

How the screen is organized

This is a schematic of the **sysadmsh** screen. Areas shown in black appear on the screen as highlighted areas or bars of text. Each area displays specific types of information:



- The Context Indicator is the highlighted bar of text in the upper-right corner of your screen. It displays the name of the current menu. The Context Indicator for the **sysadmsh** opening screen shows *SysAdmSh.*
- The Menu Line displays the menu options that are currently available. The main **sysadmsh** menu consists of ten options: System, Backups, Accounts, Printers, Media, Jobs, Dirs/Files, Filesystems, User, and Quit.
- The Description Line gives you a brief description of the currently highlighted menu option.
- The Status Line is the highlighted bar of text that separates the Menu and Description Lines from the Display Window. The Status Line in the **sysadmsh** opening screen contains the date, time, and current working directory. When a UNIX system command is executed, the name of the command and the options used are displayed briefly at the far left of the Status Line.

- The Command/Form Line displays a title for the contents of the Display Area. The title can be either a UNIX system command name or the name of a **sysadmsh** form. When a command name is displayed, the location of the manual page associated with the command is appended in parentheses. For example, when System ➪ Report ➪ Users is selected, the Command Line displays "`who C`". This means that the command can be found on the **who** manual page.
- The Display Area displays **sysadmsh** forms and scan windows. Forms and scan windows are explained in detail later in this chapter.
- Error Messages and recovery instructions appear on the last line of the screen in highlighted text.

Selecting menu items

The keyboard and mouse operations listed in Table 2.1 are used to move through the menus. Note that there are several ways to select options; if you have used menu-based programs before, use the method you are most familiar with.

Table 2-1 Basic menu operations

Keyboard	Mouse	Action
Arrow keys or ⟨Space⟩ (same as right arrow)	Move cursor to option	moves to menu option.
First letter of option, or move highlight to option and press ⟨Return⟩	Left or middle button	selects menu option.
⟨Esc⟩	Right button	retreats to previous menu.
⟨F1⟩	-	gets help.

You can familiarize yourself with the menu options by using the Arrow keys, ⟨Space⟩, or mouse to move the highlight from option to option. Each time you move the highlight to a new option, a description of that option appears on the description line.

sysadmsh has a hierarchical menu structure. Many of the menu options move you down to another menu. For example, when you select the Jobs option from the main menu, a submenu containing more options is displayed which lets you check on and manipulate your machine's processes. The menu hierarchy makes it easy to find the command you need by moving down from one menu to the next. Eventually you get to a menu option that either executes a UNIX system command or displays a form that you must fill in with the details that the command needs. Note that typing the first letter of the option name is the quickest way to move through menu levels; in time you will be able to reach the function you need instantly by pressing three- and four-letter codes you have memorized. (Table 2.9 at the end of this chapter lists all available **sysadmsh** selections and their shorthand forms.)

The best way to learn how to use menus is to practice making menu selections with the keyboard or mouse. If you select an option by mistake, you can always retreat to the previous menu by pressing the ⟨Esc⟩ key or the right mouse button. If you are several levels deep, you can return to the main menu by pressing the ⟨F2⟩ key and then typing **n**. ⟨F2⟩ takes you to the Quit option, and **n** returns you to the main menu. (The ⟨F2⟩ key does not work if you have been dropped into a UNIX system command, such as **vi**(C).) To help you find your way through the **sysadmsh** menus, Table 2.2 contains a map of the second-level menus.

Table 2-2 Map of second-level menus

System	Backups	Accounts	Printers	Media
↓	↓	↓	↓	↓
Report	Create	User	Configure	List
Configure	Restore	Defaults	Schedule	Extract
Hardware	Schedule	Terminal	Request	Archive
Software	View	Report	Auxiliary	Format
Audit	Integrity	Check	Priorities	Duplicate
Execute				Tapedump
Terminate				

Jobs	Dirs/Files	Filesystems	User	Quit
↓	↓	↓		↓
Report	List	Check		Yes
Terminate	View	Mount		No
Authorize	Copy	Unmount		
	Edit	Add		
	Modify	Floppy		
	Print	DOS		
	Archive			
	Differences			
	Remove			
	UseDOS			

When you select a menu option, one of three things happens:

- A lower-level menu is displayed.
- You are dropped into a form.
- A UNIX system command is executed and the result displayed in a scan window.

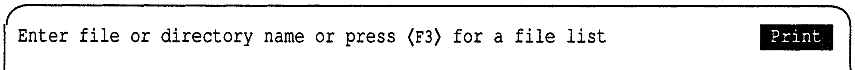
The next two sections explain forms and scan windows.

Using forms

Some menu options require additional information to perform the correct task. For example, the Print option cannot do anything until you tell it what you want to print and which printer to use. When you select this type of option, a form appears on the screen. By filling in the form, you give the command the information it needs.

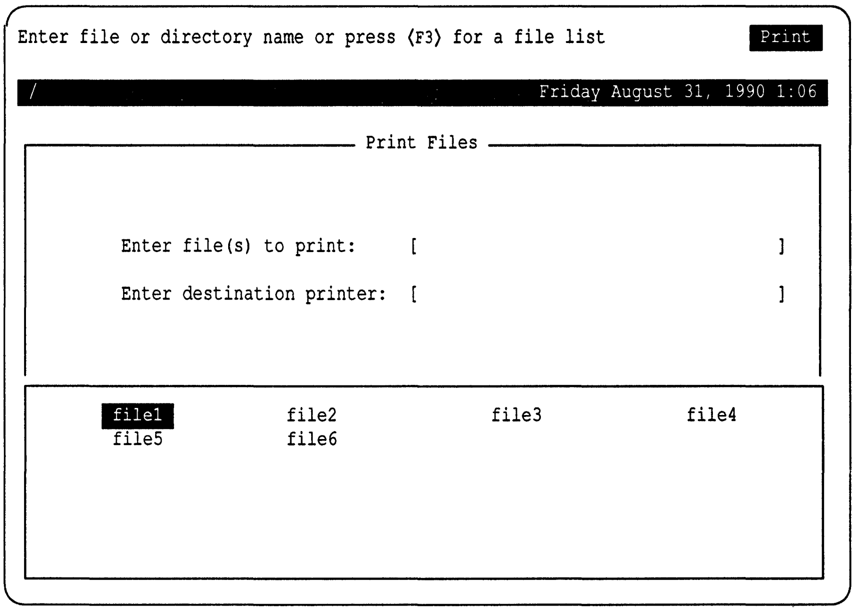
The following example demonstrates how forms work, by showing you how to print a file in your current directory. After the example, Tables 2.3, 2.4, and 2.5 list the keystrokes that allow you to move around the form, edit it, and make "point-and-pick" selections.

To print a file, first select Dirs/Files ➪ Print. The Print form is displayed:



```
Enter file or directory name or press <F3> for a file list          Print

 /                                            Friday August 31, 1990 1:06

                              Print Files

          Enter file(s) to print:       [                               ]

          Enter destination printer:    [                               ]
```

Notice that the highlight is on the first item in the form. You can fill in the field or obtain a list of choices by pressing ⟨F3⟩. You can enter the filename if you know it but, for the sake of this exercise, assume that you need to find the filename and press ⟨F3⟩ now. A window opens up overlapping part of the Print form:

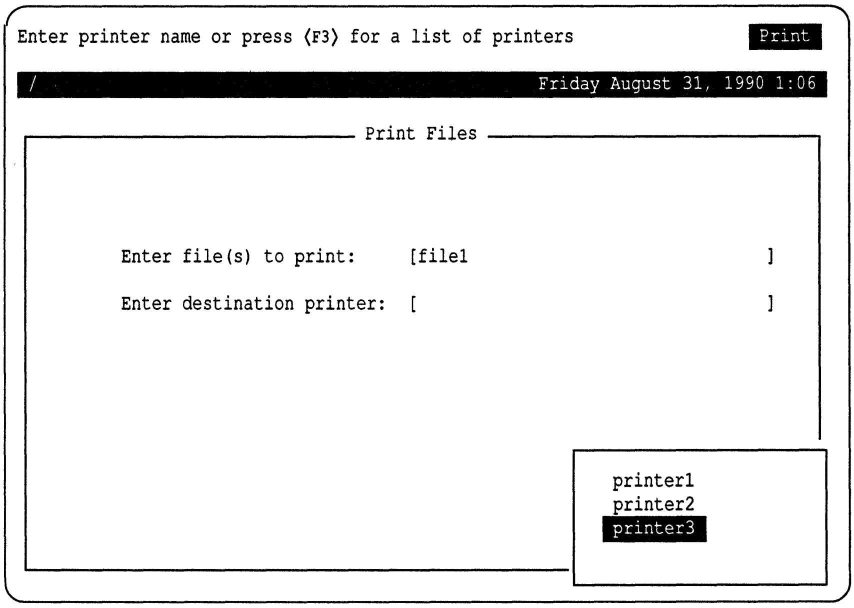


```
Enter file or directory name or press ⟨F3⟩ for a file list            Print

/                                          Friday August 31, 1990 1:06

                         Print Files

      Enter file(s) to print:      [                                  ]

      Enter destination printer:  [                                  ]

      file1          file2          file3          file4
      file5          file6
```

The window contains a list of the files that you can select. To select a file, "point" to it by highlighting it, and "pick" it by pressing return. This is known as point-and-pick, and it is used whenever a range of choices is displayed. After making your selection, the window closes and you return to the Print form.

Note that the name of the file you selected is now displayed in the form. You can now change the name using the edit keys (listed in Table 2.4 later in this section), or press ⟨Return⟩ to move to the next field.

Now enter the name of the printer to be used. If you do not know the printer name, press ⟨F3⟩. Another, smaller window opens that contains a list of installed printers:



```
Enter printer name or press ⟨F3⟩ for a list of printers                Print
/                                          Friday August 31, 1990 1:06
                              Print Files

        Enter file(s) to print:     [file1                           ]

        Enter destination printer:  [                                ]

                                                      printer1
                                                      printer2
                                                      printer3
```

You can select the printer just as you did the name of the file. After selecting a printer, you return to the Print menu.

The keystrokes listed in the following tables allow you to use forms easily.

Table 2-3 Form operations

Keyboard	Mouse	Action
⟨Esc⟩	Right button	tells the program that you changed your mind and do not want to finish filling in this form. The form is removed, and no action is performed. You are returned to the previous menu. In addition, ⟨Esc⟩ followed by ⟨Return⟩ acknowledges that an error message was read and that you are ready to continue.
Up, Down Arrow	Left button	moves to other fields in a form. Some fields are restricted and no input is allowed. The Arrow keys skip over these. Other fields must be filled in. With the mouse, move the cursor to the field and click the Left button. Pressing the Down Arrow key on the last item in a form brings you back to the first item.
Left, Right Arrow	-	moves left and right in the current field. This allows you to change text without retyping the entire line.
⟨Return⟩	Middle button*	completes the data entry to a field and moves the cursor to the next field. In the last field, pressing ⟨Return⟩ or the middle mouse button completes the entire form and tells the shell that the data is ready to use.
⟨Ctrl⟩x	-	exits and executes the form from wherever you are. Think "x" for "execute". ⟨F10⟩ does the same.
⟨F4⟩	-	calls the spelling checker utility when you are in a form. If you think a word might be misspelled, press ⟨F4⟩ while the cursor is on the word and a list of possible correct spellings appears in a point-and-pick list. The word you select replaces the misspelled word.
⟨F10⟩	-	exits and executes the form from wherever you are. ⟨Ctrl⟩x does the same.

* On a two-button mouse, pressing both buttons simultaneously emulates the middle mouse button.

Table 2-4 Edit keystrokes*

Keystroke	Action
⟨Ctrl⟩y	deletes the current line, and begins the line again.
⟨Ctrl⟩w	deletes the current word.
⟨Ctrl⟩g-⟨Ctrl⟩h	moves the cursor to the beginning of the line.
⟨Ctrl⟩g-⟨Ctrl⟩l	moves the cursor to the end of the line.
⟨Ctrl⟩v	toggles into or out of overstrike mode.
⟨Del⟩	deletes the character over the cursor.
⟨Bksp⟩	backs up and deletes one character (left of cursor).
⟨Ctrl⟩u	pages up - moves up one page.
⟨Ctrl⟩d	pages down - moves down one page.
⟨Ctrl⟩n	goes to the next word.
⟨Ctrl⟩p	goes to the previous word.
Left, Right Arrow	moves left and right within the edit line.

* You cannot use the mouse for edit operations; if a form opens a window for input, you must use the keyboard. Even the Right button (⟨Esc⟩) will not affect the window.

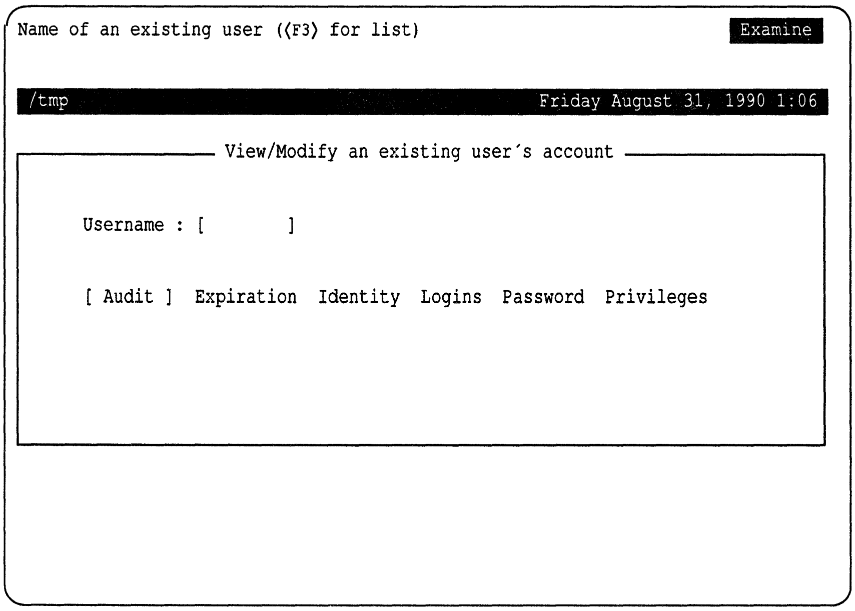
Table 2-5 Point-and-Pick Operations

Keyboard	Mouse	Action
⟨Return⟩	Middle button*	selects the item.
⟨Esc⟩	Right button	ends the selection process. The list is removed and no action is performed.
⟨Ctrl⟩v	-	toggles between selecting all or none of the items appearing in a list.
Up, Down Arrow	Move mouse up or down	moves to other items in a list.
Left, Right Arrow	Move mouse right or left	moves across a multicolumn display.
⟨Space⟩	Left button	marks items when the application accepts more than one. A marked item is indicated by an asterisk (*) in the left column. It may be unmarked by pressing ⟨Space⟩ a second time while the item is selected. The entire collection of marked items is selected by pressing ⟨Return⟩.
⟨F5⟩	-	finds items in long listing. It is called the "Search" key. A prompt appears and you enter the string to search for, then press ⟨Return⟩. If the item is found, the highlight moves to that item, and another ⟨Return⟩ selects the item. If no match is found, the highlight does not move. The ; and : keys repeat the previous search, forward and backward respectively.
First letter	-	selects an item by its first letter. It is the fastest method of selection. Pressing ⟨Return⟩ selects the highlighted item. (If there is only one item beginning with that letter, it is marked by typing its first letter. There is no need to press ⟨Return⟩ again.) If several items begin with the same letter, the cursor moves to the first occurrence in the list.

* On a two-button mouse, pressing both buttons simultaneously emulates the middle mouse button.

Using radio buttons

Radio buttons are rows of selection boxes in a form. They are selected in a way similar to items in a menu bar. The purpose of radio buttons is to provide a secondary level of selections from within a form, as in this example, taken from the Accounts menu of the **sysadmsh**:



```
Name of an existing user (<F3> for list)                    Examine

 /tmp                                     Friday August 31, 1990 1:06

 ─────────── View/Modify an existing user's account ───────────

      Username : [        ]

      [ Audit ]  Expiration  Identity  Logins  Password  Privileges
```

In this example, the "Username" field must be filled in before the radio button selection can be made. The reference syntax for **sysadmsh** includes a notation for radio buttons, which is a colon (:). Look at the following reference:

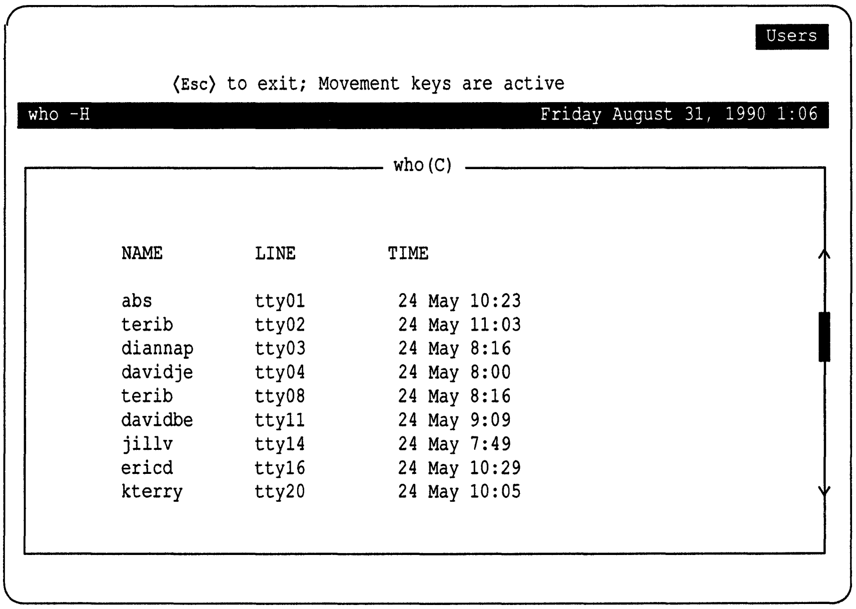
Accounts ➪ User ➪ Examine:Audit

When you see a **sysadmsh** reference like this, it means that when you enter the Examine menu you must provide information (in this case, the user name) before making the next selection: Audit.

Using scan windows

When you execute a UNIX system command by selecting a **sysadmsh** menu option, the result of the command is typically displayed in a scan window. Scan windows also display the contents of files and directory listings. To demonstrate the use of scan windows, let's say you want to know who is currently logged on to the system. To do this, select System ➪ Report ➪ Users. (This runs the UNIX system **who**(C) command.)

When you select the Current option, a scan window displaying the output of the **who**(C) command appears in the display area:


```
                                                              Users

          <Esc> to exit; Movement keys are active
who -H                                       Friday August 31, 1990 1:06

                              who(C)

      NAME        LINE        TIME

      abs         tty01        24 May 10:23
      terib       tty02        24 May 11:03
      diannap     tty03        24 May 8:16
      davidje     tty04        24 May 8:00
      terib       tty08        24 May 8:16
      davidbe     tty11        24 May 9:09
      jillv       tty14        24 May 7:49
      ericd       tty16        24 May 10:29
      kterry      tty20        24 May 10:05
```

Note that the name of the command **who** and the reference section in which its description can be found (C) are displayed at the top of the window. Also note that the option given to the command (**-H**) is displayed in the right hand side of the Status Line. If you do not understand the information displayed, look up the proper manual page for more information.

The vertical scroll bar (at the extreme right edge of the window) shows the position of the current screen relative to the whole document, and moves the screen up or down by an amount that depends on how far you move the scroll bar. When you are viewing the first screen in a document, the top arrow is covered by the highlighted block; when you are viewing the last screen in a document, the bottom arrow is covered by the highlighted block. If the document is less than one page long, the scroll bar does not appear.

To operate the scroll bar, press and hold the ⟨Down Arrow⟩ or ⟨Up Arrow⟩ keys, depending on which direction you want the screen to move over the page. When the scan window shows the area of the document that you want to view, release the key. The ⟨PgUp⟩ and ⟨PgDn⟩ keys can be used to move over the document a page at a time.

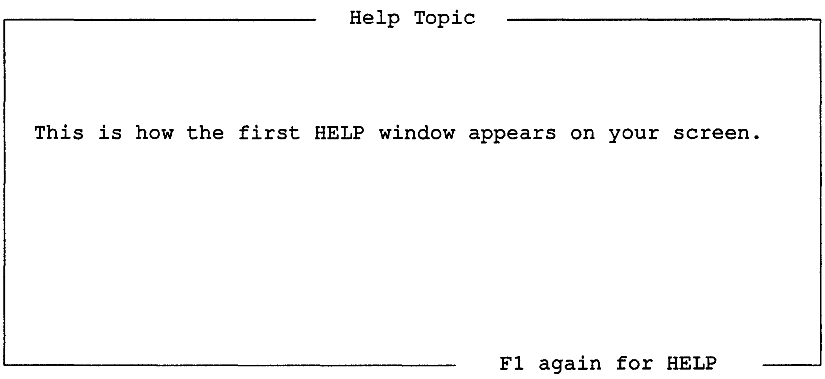
Use the keys listed in Table 2.6 when you are in a scan window.

Table 2-6 Scan operations

Keyboard	Mouse	Action
⟨Esc⟩	Right button	exits the file.
⟨Up Arrow⟩	-	moves up one line.
⟨Down Arrow⟩ or ⟨Return⟩	-	moves down one line.
⟨PgDn⟩ or ⟨Space⟩	Middle button	moves down a page.
⟨PgUp⟩	Left button	moves up a page.
⟨Home⟩	-	moves to the top of the display.
⟨End⟩	-	moves to the bottom of the display.
⟨F5⟩	-	searches for a pattern in the display. (The ; and : keys repeat the search forward and backward, respectively.)
⟨F7⟩	-	prints the output of the command or file currently in the scan window.

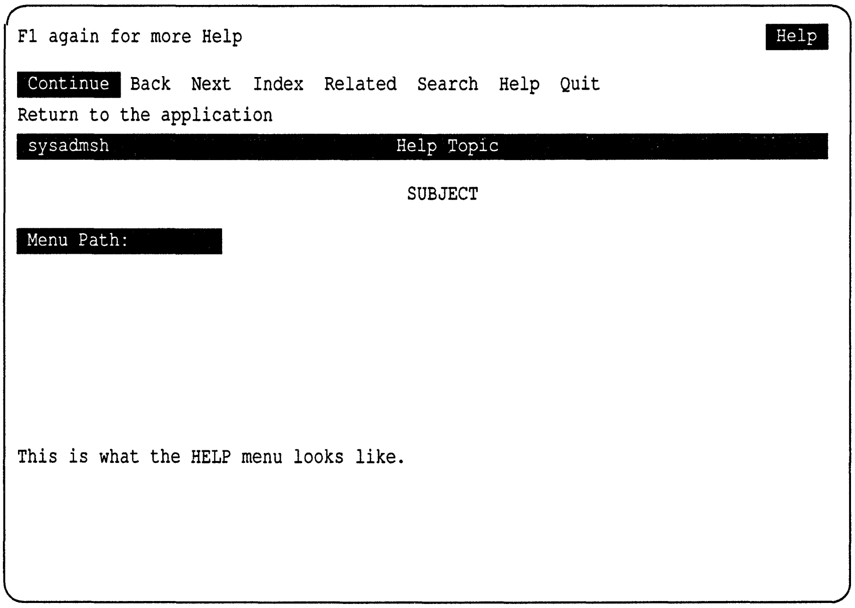
Getting help

You can press the ⟨F1⟩ key to display more information to help you with your selection. When you press the ⟨F1⟩ key, a Help window opens within your current screen. It looks like this:



```
Help Topic

This is how the first HELP window appears on your screen.

F1 again for HELP
```

The window contains some basic information. If you need more help, you can press ⟨F1⟩ again and the complete Help menu is displayed:



```
F1 again for more Help                                              Help

Continue Back Next Index Related Search Help Quit
Return to the application
sysadmsh                         Help Topic

                                  SUBJECT

Menu Path:

This is what the HELP menu looks like.
```

When you are finished, select Quit from the Help menu and you return to your place in the **sysadmsh** menu.

The menu options for Help are listed in Table 2.7.

Table 2-7 Help options

Option	Action
Continue	continues on to the next page of help text. All the vertical movement keys are active: Up and Down Arrows, Page Up and Page Down, Home and End. If there is no further information, the highlight moves to the Quit option on the Help menu and the description line reads "Return to the application".
Back	moves back to topics that were seen previously. There is no corresponding "Forward". This also backs up to more general topics. You can go back until the top-level introductory topic is reached.
Index	chooses a new topic from a list of indexed topics.
Related	chooses a new topic related to the current one.
Search	searches for a new topic by matching a pattern. First, you specify where to look (the titles, the text lines, or both), and then give the pattern. The pattern can be a simple keyword (like "create" or "date") or a more complex "regular expression". (See the *Programmer's Reference* for further information on regular expressions.) A list of topics containing the pattern is presented.
Help	explains how the help facility itself is used. A table similar to this one is displayed on the screen. If you need further information, look for your topic in Index, Related, or Search.
Quit	exits Help and returns to **sysadmsh.** ⟨F2⟩ or ⟨Esc⟩ are other ways to exit quickly.

Each Help screen has general information available, as well as specific information about each option listed on the menu from which Help was selected. Each descriptive passage is preceded by the associated menu line and followed by a reference to the operating system documentation.

NOTE When you are within a particular UNIX system command, you do not have access to the Help facility. For example, when you select Dirs/Files ➪ Edit , you are within the UNIX system **vi** command, and the **sysadmsh** keys no longer function. When you exit the command and return to the **sysadmsh**, the keys function as expected. If no element of the **sysadmsh** is visible on the screen (menu line, boxes, context indicator, and so on) then Help is probably unavailable. If you need help, exit from the current process and press the ⟨F1⟩ key to view Help. In general, it is best to use Help prior to executing a menu selection.

Changing the current directory within sysadmsh

There are many occasions when it is necessary to change your current directory to use certain files and commands. You can move to another directory by pressing the ⟨F6⟩ key. The current directory is displayed at the top of the screen. You can use the ⟨Bksp⟩ key to erase the name of the current directory (to begin again), or you can add to or alter part of the current name. When you press ⟨Return⟩, your directory change is executed and reflected on the status line.

The function keys

The function keys give you access to several time-saving features. See Table 2.8.

Table 2-8 Function keys

Key	Action
⟨F1⟩ Help key	displays help for the current context within the application. Further information is available by pressing ⟨F1⟩ again.
⟨F2⟩ Exit key	activates the Quit option on the top menu level. Press **n** to return to **sysadmsh** or **y** to exit.
⟨F3⟩ Pop-up key	(used within a form) displays a list of items that are acceptable for the current field.
⟨F4⟩ Spell key	(used within a form) displays a list of words that are possible correct spellings of the word in the current field. Select a word from the list by highlighting it, then pressing ⟨Return⟩. The word is then placed in the field. This function is only available if the **Speller** is installed.
⟨F5⟩ Search key	(used within a window) prompts for a string to search for. When you enter a string and press ⟨Return⟩, the highlight moves to the item in the list that matches the pattern. If no match is found, the search fails and the highlight does not move. In addition, the semicolon (;) repeats a search forward and the colon (:) searches backward.
⟨F6⟩ New directory key	offers the opportunity to change your current working directory. Note that this does not change the directory you will return to upon leaving **sysadmsh**
⟨F7⟩ Print key	prints the output of any command that is displayed in a scan window.

Using shell escapes to access the UNIX system command line

You can execute a UNIX system command from within a **sysadmsh** menu by typing the shell-escape character, an exclamation point or mark (!). The menus are replaced by a subshell that displays a text-entry line and a prompt asking for a command. When you enter the command and press ⟨Return⟩, the command is executed by the shell. After the command is completed, the output is displayed on the screen, and you are prompted to press any key to return to the shell.

NOTE The UNIX system command line can only be accessed from the shell menus. It cannot, for example, be accessed from a form or a point-and-pick list.

sysadmsh environment variables

The **sysadmsh** uses the following environment variables, which can be defined in user *.login* or *.profile* files:

SA_EDITOR	If not set, the default editor is Lyrix if installed, or **vi**(C) if Lyrix is not available.
SA_PRINT	If not set, **sysadmsh** pipes output to the */bin/lp* program; if set, **sysadmsh** sends output directly to the */dev/lp* printer device.
SA_USERAPPS	If not set, the default user application file is *$HOME/.sysadmmenu*

Customizing sysadmsh menus

Third party developers can customize **sysadmsh** menus as desired. See the **menumerge**(ADM) manual page for complete instructions.

sysadmsh menu options

Table 2.9 lists the options available in the **sysadmsh** menus. The main or top-level menu options appear in boldface type at the left margin of the table. The options listed below each top-level menu option are in lower levels of the menu hierarchy. Those with one arrow are in the first-level menus, those with two arrows are in the second-level menus, and so on. For instance, at the beginning of the table, Report is a first-level menu option in the System menu, and Activity is an option in the second-level Report menu.

Once you are familiar with the menu options, you can enter shorthand menu paths to reach lower level options without having to see all of the upper level menus. In Table 2.9, the "Path" column lists the shorthand menu path for each menu option. Enter the first letter of each menu option in the path down to the option you want. For example, to choose the Activity option, enter the first letters of system, report, and activity, **sra**, without pressing the ⟨Return⟩ key.

If there are no menu options listed below the one you choose, either a form or a display appears. For example, when you choose the Activity option, a scrollable display or scan window appears, showing system processes.

Table 2-9 sysadmsh menu map

sysadmsh option	Path	Function
System	s	system-wide reports, configurations
➪Report	sr	reports on current state of system
➪➪Activity	sra	reports on current system activity
➪➪Users	sru	reports users currently logged in
➪➪Printers	srp	reports on current status of printers
➪➪Disk	srd	reports on current disk usage
➪➪Network	srn	reports on current network files
➪➪➪Xnet	srnx	reports on XENIX-NET status
➪➪➪UUCP	srnu	reports on UUCP status
➪➪Messages	srm	reads system messages
➪➪Software	srs	checks installed software/package status
➪Configure	sc	configures system files
➪➪Security	scs	changes level of system security
➪➪➪Relax	scsr	resets security to default UNIX system levels
➪➪Kernel	sck	changes kernel parameters or capabilities
➪➪➪ParaMeters	sckp	configures tunable kernel parameters
➪➪➪Rebuild	sckr	relinks kernel according to current settings
➪➪➪DOS	sckd	adds DOS filesystem support to the kernel
➪➪➪Streams	scks	adds streams support to the kernel
➪➪➪Layers	sckl	adds shell layers support to the kernel
➪➪Logout	scl	sets idle time before user is logged out
➪➪Defaults	scd	modifies system default parameters
➪➪➪Home	scdh	modifies default home directory for users
➪➪➪Message	scdm	modifies message of the day file
➪➪➪Checklist	scdc	lists which filesystems checked at startup
➪➪➪Other	scdo	modifies other default files
➪➪International	sci	configures system for international use
➪➪➪System	scis	sets system locale variables
➪➪➪Individual	scii	changes a user's default locale
➪➪➪Display	scid	changes mapping of terminal character set
➪➪➪Keyboard	scik	changes mapping of console keyboard
➪➪Network	scn	configures networking files
➪➪➪UUCP	scnu	configures UUCP files, enable/disable tty
➪➪Time	sct	sets the system time
➪➪Menus	scm	customizes **sysadmsh** menus
➪➪Other	sco	executes third-party **sysadmsh** extensions
➪Hardware	sh	adds or removes hardware from the system
➪➪HardDisk	shh	adds a hard disk to the system
➪➪Tape	sht	adds or removes a tape drive from system

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
➪➪Printer	shp	adds or removes a printer from system
➪➪Card_Serial	shs	adds a serial card to system
➪➪Mouse	shm	adds mouse to system
➪➪Video	shv	configures video card graphics parameters
➪Software	ss	adds or removes software from system
➪Audit	sa	administers/examines system auditing data
➪➪Enable	sae	enables audit using existing parameters file
➪➪Disable	sad	disables audit (stops collection of process data)
➪➪Collection	sac	displays or modify audit collection rules
➪➪➪Directories	sacd	displays or modify audit directory list
➪➪➪➪List	sacdl	displays audit directory list
➪➪➪➪Create	sacdc	creates new audit directory
➪➪➪➪Delete	sacdd	deletes existing audit directory
➪➪➪➪Add	sacda	adds entry to audit directory list
➪➪➪➪Remove	sacdr	removes entry from audit directory list
➪➪➪Events	sace	displays/modifies list of audited events
➪➪➪➪View	sacev	displays system audit collection mask
➪➪➪➪Modify	sacem	modifies system audit collection mask
➪➪➪IDs	saci	displays/modifies list of users/groups audited
➪➪➪➪View	saciv	displays list of users and groups audited
➪➪➪➪Modify	sacim	modifies list of users and groups audited
➪➪➪Parameters	sacp	displays or modifies audit parameters
➪➪➪➪View	sacpv	displays audit parameters
➪➪➪➪Modify	sacpm	modifies audit parameters
➪➪➪Reset	sacr	changes collection rules to default values
➪➪➪Statistics	sacs	displays statistics of current audit session
➪➪Report	sar	reports on stored audit session data
➪➪➪List	sarl	lists report templates available
➪➪➪View	sarv	views parameters in a report template
➪➪➪Create	sarc	creates a new report template
➪➪➪Modify	sarm	modifies an existing report template
➪➪➪Delete	sard	deletes an existing report template
➪➪➪Generate	sarg	generates report of an audit session
➪➪Files	saf	manipulates audit session files
➪➪➪List	safl	lists audit session files on system

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
➪➪➪Backup	safb	backs up audit session file to removable media
➪➪➪Delete	safd	removes audit session file
➪➪➪Restore	safr	restores audit file from removable media
➪Execute	se	executes programs that are system specific
➪Terminate	st	shuts down system to remove power or reboot
Backups	b	performs backups of files
➪Create	bc	creates backups
➪➪Scheduled	bcs	performs scheduled filesystem backups
➪➪Unscheduled	bcu	performs unscheduled filesystem backup
➪Restore	br	restores filesystems and files
➪➪Partial	brp	restores specific directories and files
➪➪Full	brf	restores entire filesystem
➪Schedule	bs	modifies scheduled backup frequency
➪View	bv	views contents of a backup
➪Integrity	bi	checks the integrity of a backup
Accounts	a	controls functions of user accounts
➪User	au	alters/creates/retires user accounts
➪➪Examine	aue	sets password/ID/authorization/audit parameters
➪➪Create	auc	makes a new user account
➪➪Retire	aur	closes an existing user account
➪Defaults	ad	sets system-wide (default) parameters
➪➪Authorization	ada	views/modifies default kernel/subsystem privileges
➪➪Password	adp	views/modifies default password life/choice
➪➪Logins	adl	views/modifies default login attempt controls
➪Terminal	at	manages terminal database entries
➪➪Examine	ate	views/modifies existing terminal entry
➪➪Create	atc	makes a new terminal entry
➪➪Delete	atd	deletes an existing terminal entry
➪➪Lock	atl	locks a specific terminal
➪➪Unlock	atu	clears all locks on a specific terminal
➪➪Assign	ata	manages device name equivalences database

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
➪➪➪Examine	atae	views/modifies existing device entry
➪➪➪Create	atac	makes a new device entry
➪➪➪Delete	atad	deletes an existing device entry
➪Report	ar	reports on passwords/terminals/login activity
➪➪Password	arp	reports on accounts by password status
➪➪➪Impending	arpi	reports on accounts: passwords near expiring
➪➪➪Expired	arpe	reports on accounts with expired passwords
➪➪➪Dead	arpd	reports on accounts with dead passwords
➪➪➪User	arpu	reports on single user's password
➪➪➪Group	arpg	reports on single group's passwords
➪➪➪Full	arpf	lists all entries in password database
➪➪Terminal	art	reports on access status by terminals
➪➪Login	arl	reports login activity by user/group/terminal
➪➪➪User	arlu	reports logins of one, a range of, or all users
➪➪➪Group	arlg	reports logins of a group or range of groups
➪➪➪Terminal	arlt	reports logins of one/range/all terminals
➪Check	ac	checks contents of tcb files for errors
➪➪Databases	acd	checks consistency of subsystem databases
➪➪Password	acp	checks */etc/passwd* and */etc/group*
Printers	p	administers print system
➪Configure	pc	configures printers on print service
➪➪Add	pca	adds printer to system
➪➪Modify	pcm	modifies printer configuration
➪➪Remove	pcr	removes printer destination from print service
➪➪Default	pcd	changes system default destination printer
➪➪Parameters	pcp	modifies printer controls and parameters
➪➪Errors	pce	sets error warning notification/recovery modes
➪➪Content	pcc	specifies type of content printable on printer
➪➪Users	pcu	specifies who can use printer
➪Schedule	ps	starts/stops print service, handle requests
➪➪Begin	psb	starts print service
➪➪Stop	pss	shuts down print service
➪➪Accept	psa	allows requests for destination

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
➪➪Reject	psr	rejects requests for destination
➪➪Enable	pse	enables printers
➪➪Disable	psd	disables printers
➪Request	pr	moves or cancels requests on print service
➪➪Move	prm	moves requests between destinations
➪➪Cancel	prc	cancels requests made to print service
➪Auxiliary	pa	handles print-wheels/filters/preprinted forms
➪➪Alert	paa	sets/lists an alert for print-wheel
➪➪Filter	paf	administers filters used with print service
➪➪➪Change/Add	pafc	adds or changes filter used with print service
➪➪➪Remove	pafr	removes filter from print service
➪➪➪List	pafl	lists a description of filter
➪➪➪Original	pafo	restores original filter description
➪➪PPforms	pap	administers pre-printed forms of print service
➪➪➪Configure	papc	modifies printer settings for pre-printed forms
➪➪➪Modify	papm	adds or changes pre-printed forms
➪➪➪Remove	papr	removes pre-printed form from print service
➪➪➪List	papl	lists attributes of existing form
➪➪➪Users	papu	allows or denies user access to a form
➪➪➪Alerts	papa	modifies alert method for mounted form
➪➪➪➪Specify	papas	specifies alerting method
➪➪➪➪List	papal	lists current alert
➪➪➪➪Terminate	papat	terminates existing active alert
➪➪➪➪Remove	papar	removes alert definition
➪Priorities	pp	sets printing queue priorities
➪➪Default	ppd	sets system-wide priority default
➪➪Highest	pph	sets default highest priority level for users
➪➪Remove	ppr	removes users from specified priority level
➪➪List	ppl	lists default priority level and limits

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
Media	m	reads, copies, compares, formats floppies/ tapes
➪List	ml	lists the contents of a floppy or tape
➪Extract	me	extracts the contents of a floppy or tape
➪Archive	ma	stores files/directories/filesystems on media
➪Format	mf	formats either a UNIX or DOS floppy
➪Duplicate	md	makes a copy of floppy or tape
➪Tapedump	mt	displays physical contents of tape
Jobs	j	views/controls processes
➪Report	jr	reports on current processes (snapshot)
➪Terminate	jt	terminates currently running process (kill)
➪Authorize	ja	authorizes users to run jobs
➪➪Scheduled	jas	authorizes creating regularly scheduled jobs
➪➪➪Default	jasd	sets default authorization for scheduled jobs
➪➪➪User	jasu	allows/prohibits user-scheduled jobs
➪➪➪View	jasv	checks who can create scheduled jobs
➪➪Delayed	jad	authorizes users to create delayed jobs
➪➪➪Default	jadd	sets the default authorization for delayed jobs
➪➪➪User	jadu	allows/prohibits user from creating delayed jobs
➪➪➪View	jadv	checks who is authorized to create delayed jobs
➪➪Environment	jae	modifies delayed job environment
➪➪➪At	jaea	modifies the environment for at-controlled jobs
➪➪➪Batch	jaeb	modifies the environment for **batch**(C) jobs
Dirs/Files	d	interacts with files and directories
➪List	dl	lists files in current directory
➪View	dv	views contents of file
➪Copy	dc	copies directory or file
➪Edit	de	edits one or more files
➪Modify	dm	changes file parameters
➪➪Permissions	dmp	changes file permissions
➪➪Ownership	dmo	changes file ownership

(Continued on next page)

Table 2-9 sysadmsh menu map
(Continued)

sysadmsh option	Path	Function
➪➪Group	dmg	changes file group ownership
➪➪Name	dmn	renames or moves files
➪➪Size	dms	compacts files
➪➪Format	dmf	changes file formats
➪Print	dp	prints files
➪Archive	da	stores files
➪Differences	dd	compares two text files or directories
➪Remove	dr	removes specified files or directories
➪UseDOS	du	uses DOS utilities to manipulate DOS files
➪➪List	dul	lists DOS files in current directory
➪➪Remove	dur	removes a DOS file or directory
➪➪MakeDir	dum	creates a DOS directory
➪➪Copy	duc	copies files between DOS and UNIX systems
➪➪View	duv	displays DOS files
➪➪Format	duf	formats DOS media
Filesystems	f	checks/acts on filesystems
➪Check	fc	checks and repairs filesystem (**fsck**)
➪Mount	fm	mounts a filesystem
➪Unmount	fu	unmounts a mounted filesystem
➪Add	fa	adds appropriate info for a new filesystem
➪Floppy	ff	creates a filesystem on a floppy
➪DOS	fd	adds support for DOS filesystem
User	u	user-specific applications read from *$HOME/.sysadmmenu*
Quit	q	quits system administration shell
➪Yes	qy	leaves the sysadm shell
➪No	qn	cancels Quit command

Chapter 3
Starting and stopping the system

This chapter explains how to do the following:

- start and stop your system
- log in as the super user *(root)*
- change the system boot procedure
- use the device and system configuration information displayed at boot time.

Additional information on customizing the system startup process is found in the "Customizing system startup" appendix in this guide.

Starting the system

Starting a UNIX system requires more than just turning on the power. You must also perform a series of steps to initialize the system for operation. To start the system, you must do the following:

- Load the operating system.
- Check the filesystems (if the system was improperly stopped).
- Choose the mode of system operation.

The following sections describe each of these procedures.

Loading the operating system

The first step in starting the system is to load the operating system from the computer's hard disk.

1. Turn on power to the computer and hard disk. The computer loads the UNIX system bootstrap program and displays this message:

```
SCO System V/386

Boot
:
```

2. Press the ⟨Return⟩ key. The bootstrap program loads the operating system.

When the system is loaded, it displays information about itself and verifies that the root filesystem (that is, all files and directories) is in order and not corrupted. If a filesystem is uncorrupted and in good order, it is called "clean". If the root filesystem is clean, you can choose the mode of operation. If not, the system requires you to clean the filesystem before choosing.

Cleaning filesystems

You must clean the root filesystem if the following message is displayed:

```
fsstat: root filesystem needs checking
OK to check the root filesystem (/dev/root) (y/n)?
```

This message is displayed only if the system was not stopped properly, as described in the section "Stopping the system" later in this chapter.

Each filesystem generates a similar message. In order to work properly, the operating system requires clean filesystems. If the above message does not appear, your filesystem is clean and ready to use.

To clean the filesystem, enter **y** (for "yes") and press the ⟨Return⟩ key. The **fsck**(ADM) utility cleans the filesystem, repairing damaged files or deleting files that cannot be repaired. It reports on its progress as each step is completed. At some point, you may be asked if you wish to salvage a file. Always answer by entering **y** or **n** and pressing the ⟨Return⟩ key. For an explanation of how **fsck** works, see "Filesystem integrity" in the "Managing filesystems" chapter later in this guide.

When cleaning is complete, the system asks you to choose the mode of operation.

Choosing the mode of system operation

You may choose the mode of operation as soon as you see the message:

```
INIT: SINGLE USER MODE

Type CONTROL-d to continue with normal startup,
(or give the root password for system maintenance):
```

The system has two modes: normal operation and system maintenance. Normal operation is for ordinary work on the system. This is the mode that allows multiple users to log in and begin work. It is also known as multiuser mode. System maintenance mode is reserved for work to be done by the system administrator, and does not allow multiple users. It is also known as single user mode.

To choose normal operation, press ⟨Ctrl⟩**d**. The system displays a startup message, and you are prompted to enter the system time (see the next section). Then the system executes commands found in the */etc/rc* directories, (this includes the */etc/rc.d* and */etc/rc2.d* directories, and so forth, referred to collectively as */etc/rc* scripts) generating startup messages for the various system services, such as the printer or network services. (These scripts are described later in this chapter.) Next, the system displays the **login:** prompt. You can now log in as a normal user, as described in the "Logging in, logging out" chapter of the *Tutorial,* or as the super user.

To choose system maintenance mode, enter the super user password (also called the "root password") and press ⟨Return⟩.

NOTE The super user *(root)* password is assigned during system installation. If you do not know the *root* password, ask the administrator who installed your system.

The super user prompt " #" is displayed. The commands in the */etc/rc* scripts are not executed. (Choose system maintenance mode only if you must do system maintenance work that requires all other users to be off the system.) When you log out of system maintenance mode using ⟨Ctrl⟩**d**, the system automatically enters normal operation.

Entering system maintenance mode by shutting down first

To go from normal operation to system maintenance mode, log in as *root* and give the following command to shut down the system:

/etc/shutdown -g*n*

Δ **sysadmsh** users select: System ➪ Terminate

where ***n*** is the number of minutes until multiuser mode is stopped. After ***n*** minutes has elapsed, you are asked to confirm your choice and the system is shut down. You are then asked to press any key to reboot. When the system has rebooted, give the root password to enter single user mode.

Entering system maintenance mode directly

To go from normal operation to system maintenance mode directly, log in as *root* and give the following command:

/etc/shutdown -g2 su

The **su** indicates that you want to go directly into single user mode rather than shut the system down.

NOTE There is no **sysadmsh** equivalent for this command.

Setting the time and date

Once normal operation starts, the system asks for the correct time and date. It displays the current time and date and then the following message:

```
INIT: New run level: 2

Current System Time is Wed Nov 13 08:19:00 PST 1991
Enter new time ([yymmdd]hhmm):
```

Unless your clock battery is drained or removed, there should be no need to change the date. To leave the time and date unchanged, simply press ⟨Return⟩. If you need to change the time and date, enter the new time and press ⟨Return⟩. The new values must be entered as two or more consecutive pairs of digits, where the digits may be one or more of the following:

yy (optional)	represents the current year. It may be any two-digit value, from 70 to 99 for the years 1970 to 1999, respectively.

mm (optional)	represents the current month. It may be any two-digit value, from 01 to 12 for the months January to December, respectively.
dd (optional)	represents the current day. It may be any two-digit value, from 01 to the last day of the month.
hh	represents the current hour. It may be any two-digit value, from 00 to 23. Hours are expressed in military time, where morning hours range from 00 to 11 and evening hours from 12 to 23.
mm	represents the current minutes. It may be any two-digit value, from 00 to 59.

For example, to change the time and date to February 3, 1991 at noon, enter:

9102031200

Press ⟨Return⟩. After accepting the new value, the system then displays the new time and date:

```
Sun Feb 03 12:00:00 PST 1991
```

If you enter an incorrect value, the system prompts you to try again. If you do not enter an optional value, the current value for that item remains unchanged. If you type a new value for the year, you must also type values for the month and day. Similarly, if you type a new value for the month, you must type a value for the day.

The time and date display is followed by service startup messages and the **login:** message.

Checking the security databases

Each time your system is rebooted (and after **fsck** is run if your system was brought down unexpectedly) the system automatically checks critical security database files. The messages appear as follows:

```
Checking tcb ...
Checking auth database ...
Checking protected subsystems database ...
Checking ttys database ...
```

This checking is done to avoid problems with gaining access to your system. In the rare case where a file is missing, you are alerted to this fact and asked to restore the file from backups.

When the system is halted suddenly by power or hardware failures, some filesystem damage can occur. Such damage can result in the removal of security database files, or can leave these files in an interim state if they were being updated at the time of the system crash. Whenever a reboot occurs, the system runs a series of programs to check the status of the database files. When the system terminates abnormally and is rebooted, this check is performed after **fsck**(ADM) is run on the root filesystem, and before entering multiuser mode. This check proceeds as follows:

1. The script */etc/smmck* (system maintenance mode checker) runs the **tcbck**(ADM) program to clean up any database files that were left in an interim state while being updated.

   When a security database file is updated, the contents of the old file (*file*) is copied or updated to create the new *-t* file (*file-t*). Next, the old file (*file*) is moved to a *-o* file (*file-o*), and the new file (*file-t*) is moved to the original name (*file*). When this process is interrupted, *-o* and *-t* files are left and must be reconciled before the system will function properly. **tcbck** first resolves any *-t* and *-o* files left in */etc/auth/system*, */etc/auth/subsystems*, */tcb/files/auth/** directories, and the */etc/passwd* and the */etc/group* files. If there are multiple versions of a file, the extra files must be removed. This is done automatically as follows:

   A. If *file*, *file-o*, and *file-t* exist and *file* is not zero length, then *file-t* and *file-o* are removed.

   B. If *file* and *file-t* exist then *file-t* is removed.

   C. If only *file-t* exists, then it is moved to *file*.

   D. If only *file-o* exists, then it is moved to *file*.

   If scenario C occurs, a message similar to the following is displayed:

   ```
   /etc/tcbck: file file missing, saved file-t as file
   ```

   This is done because the *-t* file is the modified version of the original file and could have been damaged; it is likely that this file does not contain all the entries of the original. This message is repeated for all files found in that state in the specified directories. (The *-o* files are not suspect because they are the original versions of the files renamed prior to updating.)

   WARNING If you do not have backups and the files */etc/group* and */etc/passwd* were regenerated by moving the *-t* files, do not restore the original files from the UNIX system distribution. The *-t* files will have most (if not all) of your entries, and the distribution versions will not have any.

2. Next, **tcbck** checks that key system files are present and that they are not of zero length. If a file is missing (or zero length), then a message similar to that shown below is displayed:

   ```
   /etc/tcbck: file file is missing or zero length
   ```

   This process is repeated for each of the following files:

   /etc/auth/system/default†
   /etc/auth/system/files
   /etc/auth/system/devassign
   /etc/auth/system/authorize†
   /tcb/files/auth/r/root†
   /etc/group
   /etc/passwd†

   When this process is complete, if any files were missing, or empty *-t* files were substituted for real files, the following message is displayed:

   ```
   /etc/smmck: restore missing files from backup or distribution.
   ```

   NOTE Corrupted files are not detected by **tcbck**, but other error messages are displayed; these messages are described in "Resolving security-related error messages" in the "Troubleshooting your system" chapter of this guide.

3. If critical database files are removed or corrupted (files marked with a dagger), then the system enters maintenance mode automatically without asking for the root password. (While this might seem like a security breach, remember that the system itself must be under lock and key or it is not secure anyway.) The messages appear as follows:

   ```
   INIT: SINGLE USER MODE
   Security databases are corrupt.
   Starting root shell on console to allow repairs.
   Entering System Maintenance Mode
   ```

   If no critical database files are missing, you are prompted to choose system maintenance mode or normal operation.

4. If you find that files must be restored, your first option is to restore the files from your backups. The second option is to restore the files from the UNIX system distribution media.

For example, if the system reported that the file */etc/group* was missing and you had a backup of the root filesystem that was created using **sysadmsh**(ADM), you would use the Backups ➪ Restore ➪ Partial selection and restore the file. If backups are unavailable, you will have to use the distribution files.

NOTE If you must restore the file */tcb/files/auth/r/root* from your distribution, retrieve volume N2 and insert it into your floppy drive, then enter the following commands:

```
cd /tcb/files/r
mount -r /dev/install /mnt
cp /mnt/tcb/files/r/root .
```

5. Use **custom**(ADM) to restore any missing files. You can restore only one file at a time; do not try to specify more than one file. When you have restored all the necessary files, exit **custom** and press ⟨Ctrl⟩**d**. You are then prompted to enter system maintenance mode or normal operation. Press ⟨Ctrl⟩**d** again.

6. **tcbck** then removes the files */etc/auth/system/pw_id_map* and */etc/auth/system/gr_id_map* because the modification times of these files are compared with those of */etc/passwd* and */etc/group*, and problems can occur when the system clock is reset. **tcbck** then tries to rebuild the map files using **cps**(ADM). If this fails then either the File Control database (*/etc/auth/system/files*) is missing, or the the File Control database entry for "/" is missing, or there are syntax errors in */etc/passwd* or */etc/group*.

7. After the system enters multiuser mode ("INIT: New run level: 2" is displayed) and you are prompted to set the system clock, */etc/authckrc* is reinvoked. If any missing files are found, warnings similar to the ones shown previously are displayed, followed by the message shown below:

   ```
   /etc/tcbck: file file is missing or zero length
   /etc/authckrc: Log in on the OVERRIDE tty and restore
   the missing files from a backup or the distribution disks.
   ```

   This means that you missed some files earlier. These files will have to be replaced when the system comes up in multiuser mode and you are allowed to log in. Write down the names of the missing files at this stage.

8. Next, the following message is displayed:

   ```
   Checking auth database ...
   ```

   The **authck**(ADM) program is run to make certain that all users listed in */etc/passwd* have Protected Password database entries. If any are missing, they are created as needed.

9. Next, the following message is displayed:

   ```
   Checking protected subsystem database ...
   ```

   The **authck**(ADM) program is run. The Protected Subsystem database files are checked to ensure that they correctly reflect the subsystem authorization entries in the Protected Password database. Each name listed in each subsystem file is verified against the Protected Password entry with the same name, so that no authorization is inconsistent between the files. Also, each Protected Password entry is scanned to verify that all the privileges listed are reflected in the Protected Subsystem database. If any inconsistencies are found, you are asked if you want them fixed automatically:

   ```
   There are discrepancies between the databases.
   Fix them (Y or N)?
   ```

   The error messages are found in the **authck**(ADM) manual page.

   NOTE If the system is autobooting (AUTOBOOT=YES appears in */etc/default/boot*), then **authck**(ADM) is called noninteractively. Warnings are displayed about inconsistencies found but **authck** is not given the opportunity to fix them. The transition to the multiuser operation then proceeds as normal.

10. Next, you see the following message:

    ```
    Checking ttys database ...
    ```

    ttyupd(ADM) is run to ensure that all ttys in */etc/inittab* have entries in the Terminal Control database (*/etc/auth/system/ttys*).

11. The system is now up and ready for logins. If any files were reported missing, you must now log in on the override terminal to restore them, following the same procedure outlined earlier. By default, the override terminal is defined as tty01, also known as the first multiscreen. If you removed the default entry in */etc/default/login*, you will have to shut the system off, reboot and enter single-user mode, and restore the files that way. When you log in on the override tty, the following message is displayed:

    ```
    The security databases are corrupt.
    However, root login at terminal tty01 is allowed.
    ```

Logging in as the super user

Many system maintenance tasks, when performed during normal operation, require you to log in as the super user. For example, you must be logged in as the super user to stop the system.

To log in as the super user, you must know the super user password. You also need to see the **login:** message on your terminal's screen. If you do not see this message, press ⟨Ctrl⟩**d** until it appears.

NOTE The super user *(root)* password is assigned during system installation. If you do not know the *root* password, ask the administrator who installed your system.

To log in as the super user, follow these steps:

1. When you see the **login:** message, enter the super user login name:

   root

   Now press the ⟨Return⟩ key. The system prompts you for the super user password.
2. Enter the super user password and press the ⟨Return⟩ key. The system does not display the password as you enter it, so enter each keystroke carefully.

The system opens the super user account and displays the message of the day and the super user prompt "#".

Take special care when you are logged in as the super user. In particular, you should be careful when deleting or modifying files or directories. This is important because the super user has unlimited access to all files; it is possible to remove or modify a file that is vital to the system. Avoid using wildcard designators in filenames and keep track of your current working directory.

You can leave the super user account at any time by pressing ⟨Ctrl⟩**d**.

Stopping the system

Stopping a UNIX system requires more than just turning off the computer. You must prepare the system for stopping by using either the **shutdown**(ADM) or (under certain conditions) the **haltsys**(ADM) command. The following sections describe each command.

Using the wall command

Before stopping the system with the **shutdown**(ADM) command, you should notify users of the impending shutdown. You may want to include other details, such as when the system will be restarted.

To send a system-wide message to the terminals of all the users who are currently logged in, use the **wall**(ADM) ("write to all") command:

```
wall
```

Press ⟨Return⟩. Enter the message, pressing ⟨Return⟩ to start a new line if necessary. When you have finished entering the message, press ⟨Ctrl⟩**d**. This displays your message on all system terminals.

Using the shutdown command

The **shutdown** command is the normal way to stop the system and should be used whenever the system is in normal operation mode. It warns other users that the system is about to be stopped and gives them an opportunity to finish their work. The warning message that **shutdown** displays at all terminals can be customized. (If desired, the system administrator can also use the **wall**(ADM) command to send a message about the impending shutdown prior to running the actual **shutdown** command.)

To stop the system with the **shutdown**(ADM) command, follow these steps:

1. Log in as the super user. See the section "Logging in as the super user" earlier in this chapter. The system opens the super user account and displays the message of the day and the super user prompt.
2. Enter the following command and press ⟨Return⟩:

   ```
   /etc/shutdown -gn
   ```

   Δ **sysadmsh** users select: System ⇨ Terminate

   where *n* is the number of minutes before the shutdown is to take place. The system displays a warning message at each terminal, asking logged-in users to finish their work and to log out. As soon as all users are logged out or the specified time has elapsed, the system closes all accounts and displays the following message:

   ```
   ** Safe to Power Off **
             -or-
   ** Press Any Key to Reboot **
   ```

3. Turn off the computer or press any key to reboot the system.

Using the haltsys command

The **haltsys**(ADM) command halts the system immediately. This command should be used only when in single user mode. If there are any users logged into the system when the **haltsys** command is given, they are logged out and their work in progress is lost. In addition, network servers and other programs are terminated abnormally and could create problems when they are restarted.

To stop the system with the **haltsys** command, follow these steps:

1. You should be in single-user mode (use **who -r** to check). Remember that the **haltsys** command should not be used in multiuser mode.
2. Enter:

   /etc/haltsys

   Now press the ⟨Return⟩ key. The system displays the following message:

```
** Safe to Power Off **
         -or-
** Press Any Key to Reboot **
```

3. Turn off the computer, or press any key to reboot the system.

Understanding the boot display information

At boot time, a table of hardware information is always displayed after the copyright information. This table represents your hardware configuration as recognized by the operating system. Example 3-1 is an annotated version of the boot screen as it appears on a sample machine.

Example 3-1 Sample boot display

```
device    address       vector dma comment
--------------------------------------------------------------------------
fpu       -             35     -   type=80387
floppy    0x03F2-0x03F7 06     2   unit=0 type=96ds15
serial    0x02F8-0x02FF 03     -   unit=1 type=Standard nports=1
parallel  0x0378-0x037A 07     -   unit=0
console   -             -      -   unit=ega type=0 12 screens=68k
disk      0x01F0-0x01F7 36     -   type=W0 unit=0 cyls=791 hds=16 secs=48
```

This information is explained in the following key:

device, address, vector, dma, comment	The name of the hardware, address in hexadecimal, interrupt vector, direct memory access channel, other details about the hardware, respectively.
fpu	floating-point unit present, specifically the Intel 80387 and 80486 chips
floppy	high density 5.25" floppy drive
serial	this is COM 1; COM 1 has one port (no multiport card is installed)
parallel	this is parallel port lp0
console	The console has an EGA video adaptor compatible with (type 0) the IBM EGA design
disk	Western Digital st506 controller number 0 (W0), hard drive 0 (unit 0), as well as the number of cylinders, heads, and sectors

The **hwconfig**(C) utility displays or accesses this information at any time, using the configuration information stored in the file */usr/adm/hwconfig*. Refer to the **hwconfig**(C) manual page for more information.

In addition, the **eisa**(ADM) utility can be used to list the cards installed in EISA machines, and the **slot**(ADM) utility used for MCA machines.

Changing the boot process

Each time the computer is started, the system runs the **boot** program. Unless you give different instructions at the prompt, **boot** loads the default kernel program using the configuration values specified in the file */etc/default/boot* on the default root filesystem. You can edit the */etc/default/boot* file to change the default configuration values for future boot operations. You can also set certain options in */etc/default/boot* to allow the system to boot automatically.

Changing the /etc/default/boot file

The **boot**(HW) manual page describes the default boot options that you can change by editing the */etc/default/boot* file.

To change which program is loaded by default when you enter only a ⟨Return⟩ at the boot prompt, modify the default bootstring set with the **DEFBOOTSTR** option in */etc/default/boot*. For example, the following setting in */etc/default/boot* causes the **boot** program to load the kernel from a hard disk by default.

```
DEFBOOTSTR=hd(40)unix
```

The first two letters of the argument specify the device (hd for hard disk or fd for floppy disk). The number in parentheses is the minor device number (40 for the root filesystem on the hard disk). Following the parentheses is the pathname of the program to be loaded.

The *Installation Guide* and the **boot**(HW) manual page describe certain keywords or "bootstrings" that you can add to the **boot** command line to load special drivers at boot time. You can also add these to the end of the default bootstring set with **DEFBOOTSTR**. For example, here is the bootstring for a Wangtek cartridge tape:

```
DEFBOOTSTR=hd(40)unix ct=wangtek(0x338,5,1)
```

Booting automatically

The settings of the **AUTOBOOT** , **TIMEOUT** , and **PANICBOOT** options in */etc/default/boot* control if, and when, automatic booting occurs.

If **AUTOBOOT=NO**, the **boot** program waits indefinitely for a response to the prompt. You can set **AUTOBOOT=YES** to allow the system to boot automatically if no response is given at the prompt after a certain amount of time. The amount of time **boot** waits for a response before booting automatically is 60 seconds by default, but you can set the number of seconds to wait with the **TIMEOUT** option. If a timeout occurs, **boot** behaves as though you entered a ⟨Return⟩ to the boot prompt, performing the default boot process using the configuration values specified in */etc/default/boot.*

You can set the **PANICBOOT** option to **YES** or **NO** to indicate whether or not the system should reboot after a panic (a panic always causes the system to halt). Refer to the **autoboot**(ADM) manual page for more information.

RAM error correction code (ECC) checking (Corollary and compatibles only)

Some Corollary and Corollary-compatible systems use memory Error Correction Code technology. An ECC daemon, or background program, scans RAM checking for single-bit errors. Single-bit errors themselves are harmless and are automatically corrected by hardware. However, if an additional bit is corrupted at the same location, a double-bit error occurs and the system panics. This is extremely rare and will probably not occur unless your RAM has developed a fault.

The ECC daemon helps avoid double bit errors by informing the system administrator of existing single bit errors. Errors are reported via the system console and */usr/adm/messages*. The system administrator should periodically check */usr/adm/messages* for any single bit error notifications and use the **ecc**(ADM) utility to map the affected 4K page out of memory.

You should configure the ECC daemon to check memory at least once per day (the default behavior).

To configure the ECC daemon, proceed as follows:

1. Log in as *root* and enter the following command:

   mkdev eccd

2. The following menu appears:

```
ECC Daemon Initialization Program

    1. Install ECC Daemon.
    2. Remove ECC Daemon.

    h. Help.
Select an option or enter q to quit:
```

You are offered the following options:

- Option 1 installs the ECC daemon.
- Option 2 removes the ECC daemon.
- Option h displays a help message.

Select the option that you require.

3. If you select option 2, the following message appears:

```
Are you sure? (y/n)
```

   Enter **y** and press ⟨Return⟩ to continue removing the ECC daemon. Enter **q** and press ⟨Return⟩ to exit **mkdev eccd**. You are returned to the root prompt. Enter **n** and press ⟨Return⟩ to abort removing the ECC daemon. You are returned to the **mkdev eccd** menu.

4. If you select option 1, you see the following message:

```
 1.  Begin memory page scan
 2.  Check for single-bit errors

Select an option or enter q to quit:
```

   The script for running the ECC daemon is created based on the options selected from this menu.

5. If you select **1**, you see:

```
Enter period, in hours, to begin memory page scan
(default 24) or enter q to quit:
```

   Enter the period for running the memory page scan and press ⟨Return⟩. If you plan to use the default of 24 (scan once a day), just press ⟨Return⟩.

6. If you select **2**, you see the following message:

```
Enter period, in hours, to check for single-bit errors
(default 10) or enter q to quit:
```

   Enter the period for checking for single-bit errors and press ⟨Return⟩. If you plan to use the default of 10 (check every ten minutes), just press ⟨Return⟩.

7. When you have finished configuring the daemon, enter **q** to begin the configuration. You see the following message:

```
Creating /etc/idrc.d/ecc...
Creating /dev/ecc...
```

   The daemon runs on the schedule specified, and is restarted automatically after a shutdown or reboot.

Checking for ECC errors

The administrator should check periodically for memory ECC errors. The ECC errors are displayed on the console and stored in */usr/adm/messages*. The messages appear as follows:

```
found a single-bit error
board=n bad_addr=xxxx
```

where ***n*** is the board number and ***xxxx*** is the address of the error. Double-bit errors are also logged and added automatically to the bad page table. So-called "hard" errors are repeated, while "soft" errors will only occur once. You should enter all hard errors into the bad page table.

Updating the bad page table with ecc(ADM)

To update the the bad memory page table, use the **ecc**(ADM) utility as described below:

1. Log in as *root* and enter the following:

   ecc

2. You see the following menu:

```
          1. Print Current Bad Memory Page Table
          2. Add Entries to Current Bad Memory Page Table
          3. Delete Entries from Current Bad Memory Page Table
          4. Induce a Single/Double Bit Error into Memory Pages

Enter your choice or 'q' to quit:
```

3. Enter **2** and press ⟨Return⟩.
4. You see the following message:

```
Enter bad memory pages.
Numbers are in decimal.
Terminate each entry with a <RETURN>.
Terminate list of bad memory pages with a 'q'<RETURN>.
Enter in any order.
This system contains pages ranging from 0 to xxx
```

   Enter the page number(s) exactly as they appear in */usr/adm/messages*.

You should reboot the system after updating the bad page table so that the change will be effected.

If you replace memory at some point, you should use option 3 to update the table and remove the affected entries. For more information, see the **ecc**(ADM) manual page.

Chapter 4

Administering user accounts

User accounts help the system administrator keep track of the people using the system and control their access to system resources. Accounts also help organize user files and protect them from access by other users. Each account has a unique "login name" and "password" with which the user enters the system, and a "home directory" where the user works. In addition, the system has certain defaults that define how long a user password should last, whether users are allowed to choose their own passwords, and how many unsuccessful login attempts should be allowed before locking a user out.

It is the system administrator's job to create accounts for all users on the system, and to maintain these accounts by changing user passwords, login groups, and other account parameters when necessary.

There is the option to have several system administrators, each charged with limited control over certain aspects of system operation, or a single *root* user controlling all administration operations (*root* exists with full super user powers by default).

This chapter covers the following topics:

- account information: for those who are familiar with UNIX systems, this section explains how database files such as */etc/passwd* and */etc/shadow* are used and how they coexist with the additional C2-related databases
- account management: how to add, alter, and remove (or retire) user accounts, plus create user groups
- default account configuration: how to configure system default login and password parameters, and if desired, change the security scheme

It is important to examine the default account restrictions soon after creating user accounts. These are summarized in "Default account configuration" later in this chapter. You should determine if these defaults are appropriate to the needs of your system.

How account information is stored

One of the principal differences between the various UNIX systems is in how account information is stored. This affects the migration of accounts across different UNIX systems, and governs how programs access this data. The files fall into two categories: UNIX system files (those defined in the System V Interface Definition) and the Trusted Computing Base (TCB) files that extend System V security. The following files are supported and maintained by the system to ensure compatibility with other UNIX systems (files marked with an asterisk are TCB files):

- */etc/passwd* file. This publicly readable file is present on most UNIX systems and contains both account data (user ID number, login shell, and so forth) and (on some systems) an encrypted account password. Password aging information is also supported. It can be edited by experienced administrators, but **sysadmsh** is the preferred method for adding and maintaining user accounts. If your system is configured with the Improved or High security defaults and you edit */etc/passwd* manually, you must run the **authck**(ADM) command with the **-p** and **-y** options to update the Protected Password database. In the lower modes, you can edit */etc/passwd* without running **authck**.
- */etc/shadow* file. This file is readable only by *root*. It contains the encrypted password otherwise found in the */etc/passwd* file. (When the */etc/shadow* exists, */etc/passwd* contains an "x" in the password field.) This file is manipulated via the **pwconv**(ADM) and **pwunconv**(ADM) utilities. This file exists by default in all security defaults except Low, where it still can be created using **pwconv**.
- */etc/default/passwd* and */etc/default/login* files. These contain default account information that, in many cases, is duplicated in the Protected Password and System Defaults database.

* Protected Password database (*/tcb/files/auth/[a-z]/username*). This database implements the requirements for the C2 level of trust as defined by the *Trusted Computing System Evaluation Criteria* (TCSEC). It contains the encrypted password of the user as well as their privileges (authorizations), password parameters, and other detailed information. The format of this file is described in the **authcap** manual page.

* System Defaults database (*/etc/auth/system/default*) file. This contains the information used for creating default accounts; the contents of this file are determined by the security defaults selected (Low, Traditional, Improved, or High). The format of this file is described in the **authcap**(F) manual page.

All database files are updated automatically when a change is made via **sysadmsh**.

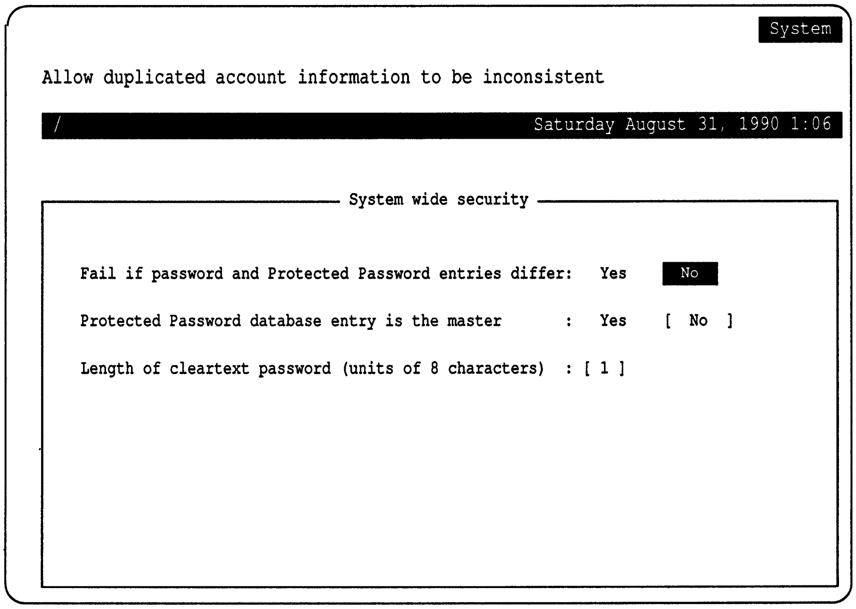
In the event of a discrepancy between these files, either the UNIX System V files or the TCB databases are used as the master to bring them into agreement. In the Low and Traditional security defaults (see "Default account configuration"), the UNIX System V files are the master. You can also configure which set of files is used as the master set; this is described in the next section.

Configuring database recovery and precedence

To define the recovery scheme to be used when a database failure occurs, make the following **sysadmsh** selection:

Accounts ➪ Defaults ➪ System

The following screen is displayed:



```
                                                                 System

Allow duplicated account information to be inconsistent

 /                                              Saturday August 31, 1990 1:06

 ----------------------------- System wide security ----------------------------

   Fail if password and Protected Password entries differ:  Yes     No

   Protected Password database entry is the master      :  Yes   [ No ]

   Length of cleartext password (units of 8 characters)  : [ 1 ]
```

The following parameters define database redundancy and precedence:

Fail if password and Protected Password entries differ

If Yes, this enforces consistency in account information; an error message is generated when an inconsistency occurs and no action is taken. If No, inconsistent data is permitted (and automatically corrected) between the different database files. Under the Improved and High security defaults, users are locked out of the system when a discrepancy occurs until the administrator logs in on the override terminal to fix the problem.

Protected Password database entry is the master

If Yes, this ensures that the trusted database files are used as the master files and information is written to the traditional UNIX system data files only for consistency — it is never relied upon for data used by the system. If No, the traditional System V data files (*/etc/passwd* and so forth) are used as the master files and the trusted databases (Protected Password and System Default) are written to only for consistency and are not relied upon for data used by the system. Under the Low and Traditional security defaults, the System V files are used as the master.

Length of cleartext password

Because the password encryption schemes on most other XENIX and UNIX systems support a maximum password length of eight characters (pre-encryption), this field provides compatibility by controlling the maximum length of passwords. The term "cleartext" refers to an unencrypted password, so password length in this instance is expressed in units of eight characters. Under the Low and Traditional security defaults, this field has a value of 1, which maintains compatibility with other UNIX systems (meaning that the */etc/passwd* file can be imported to other systems).

Account management

This section explains how to create and manage user accounts.

Adding a user

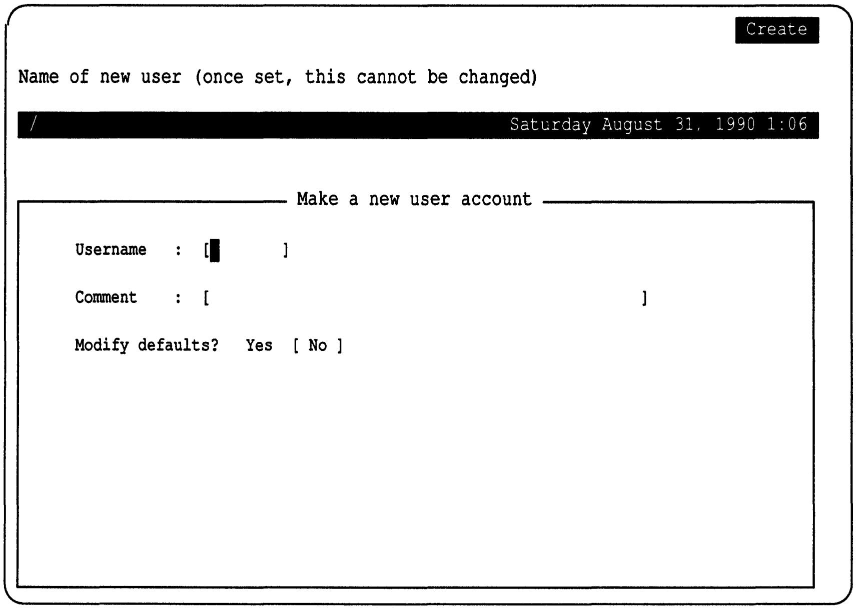
You can add a user account to the system with **sysadmsh**(ADM), which creates a new entry in the Accounts database. The database contains information about the new user (such as login name and initial password) that the system uses to let the user log in and begin work. **sysadmsh** also creates a home directory for the user, a mailbox for use with the **mail** command, and an initialization file (for example, *.profile* for the Bourne and Korn shells or *.login* for C-shell) containing UNIX system commands that are executed when the user logs in.

NOTE Although **sysadmsh** is the preferred method for adding and maintaining user accounts, experienced administrators can edit the */etc/passwd* file directly. If your system is configured with the Improved or High security defaults and you edit */etc/passwd* manually, you must run the **authck**(ADM) command with the **-p** and **-y** options to update the Protected Password database. In the lower modes, you can edit */etc/passwd* without running **authck**.

To create a user account, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Create

The following screen is displayed:



```
                                                              Create
Name of new user (once set, this cannot be changed)

 /                                       Saturday August 31, 1990 1:06

 ------------------ Make a new user account ------------------

     Username   :  [          ]

     Comment    :  [                                 ]

     Modify defaults?   Yes  [ No ]
```

Follow these steps to add a user:

1. Fill in the "Username" and, if desired, "Comment" fields.
2. If you wish to alter the defaults, select Yes and define the fields as shown in the next section, "Modifying account creation defaults." Fill in each field as necessary; press ⟨F3⟩ to choose from point-and-pick lists. When you press ⟨Return⟩, the field is filled in with the value you selected.
3. When you exit the form, a window pops up to confirm your additions. If confirmed, a series of creation messages are displayed that look like this:

   ```
   Created home directory: pathname
   Created shell file: filename
   Greetings mail sent to user: name
   ```

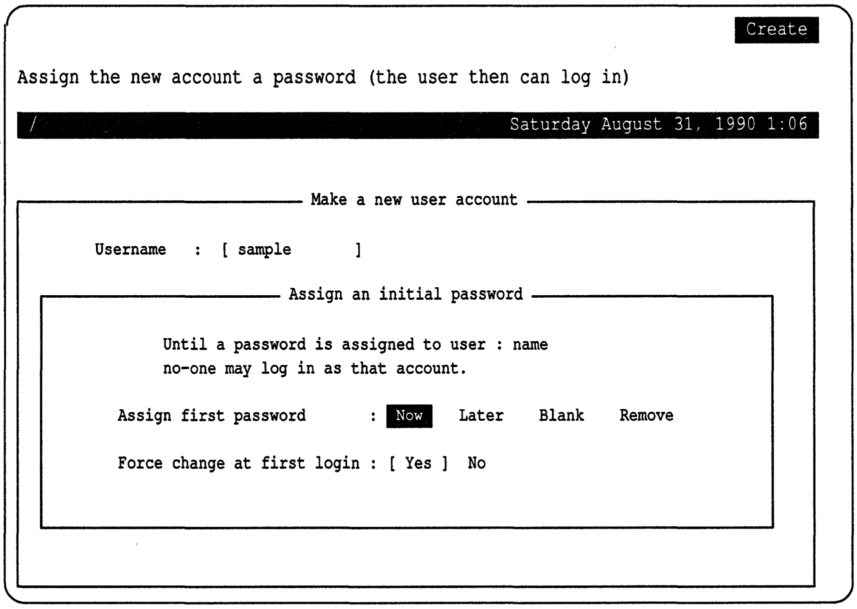
   This indicates that all the necessary files and directories were created. (This default information is taken from */usr/lib/mkuser*.)

NOTE If you see a message similar to the following:

```
useshell: Warning
useshell: File Control database inconsistency
useshell: Script path /usr/lib/mkuser/mkuser.init may be compromised
          - run integrity(ADM)
```

This means there may be a problem with the permissions of the database file indicated (including any part of the path). This can be remedied by using the **fixmog**(ADM) utility to correct permissions on the system (use **fixmog -i** to fix any problems interactively). We recommend that you run the **integrity**(ADM) utility first to get a report of all incorrect permissions on the system, but note that **integrity** does not make changes.

4. Finally, **sysadmsh** prompts you as to whether an initial password should be created. The following menu is displayed:



```
                                                               Create

Assign the new account a password (the user then can log in)

/                                          Saturday August 31, 1990 1:06

──────────────────── Make a new user account ────────────────────

      Username  :  [ sample      ]

  ─────────────── Assign an initial password ───────────────

          Until a password is assigned to user : name
          no-one may log in as that account.

       Assign first password       :  Now   Later   Blank   Remove

       Force change at first login : [ Yes ]  No
```

The possible selections for "Assign first password" are as follows:

Now	assigns the new account a password
Later	does not assign the new account a password (the user cannot log in)
Blank	assigns the new account an empty password (the user is asked to set a password at first login)
Remove	assigns the new account no password at all (the user can log in without a password)

The "Force change at first login" field governs whether users must change their passwords the first time they log in. Note that setting this to Yes negates the effect of Remove; that is, the user is required to set a password when logging in.

5. If you selected to generate a password for the new user, you see the following prompt:

```
Setting password for user user
Password is forced for user

                  Choose password

You can choose whether you pick your own password,
or have the system create one for you.

         1. Pick your own password
         2. Pronounceable password will be generated for you

Enter choice (default is 1):
```

6. If you select 1 and you are operating under the Improved or High security defaults, you see the following prompts:

```
Please enter new password (at least 5 characters):
Please choose a password which contains a mixture of lower-
and upper-case letters, digits (0 - 9), and non-alphanumeric
characters (e.g., !, #, @, ;, %, or /.)

Please do NOT choose a password that is an English word,
or which is the name of a person, place, or thing, or which
contains the string "SCO", "XENIX", or "UNIX" (in either case).
Re-enter password:
```

Note that the password is not displayed on the screen as you enter it. You are asked to enter another password if the one you entered is unacceptable.

7. If you select 1 and you are operating under the Low or Traditional security defaults, you see the following prompts:

```
Please enter new password (at least 1 character):

New password:
```

You can also simply press ⟨Return⟩ to establish a null password, where the user presses ⟨Return⟩ when prompted for the password.

8. If you select 2, the following is displayed:

```
Generating random pronounceable password for user.
The password, along with the hyphenated version, is shown.
Hit (Return) or <ENTER> until you like the choice.
When you have chosen the password you want, type it in.
Note: Type your interrupt character or 'quit' to abort at any time.

Password:xxxxxxxx  Hyphenation:xx-xx-xx Enter password:
```

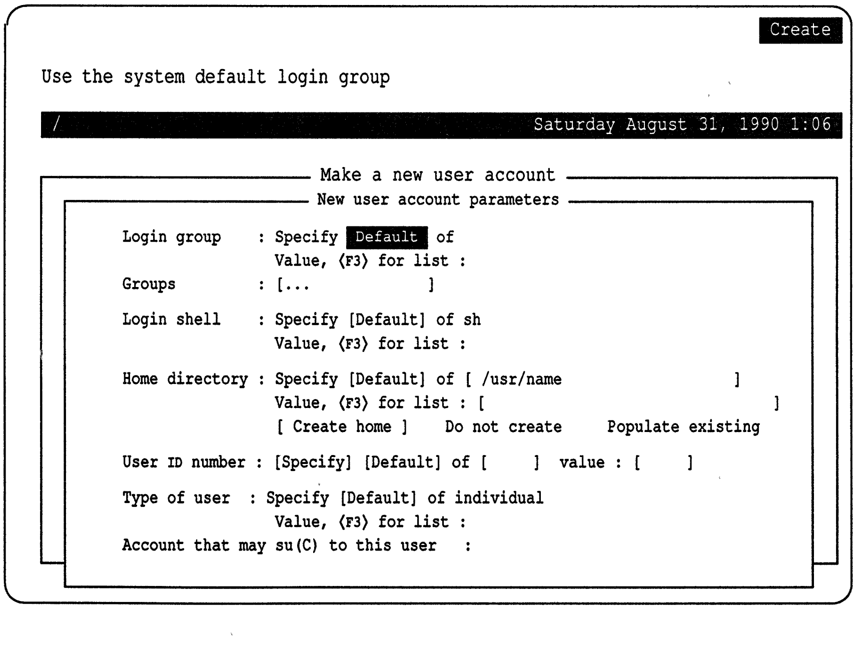
   The generated password is displayed with a hyphenated version. The hyphenation separates the password into pronounceable syllables and is designed to help you commit the password to memory.

9. Give the new password to the user. If you selected to force a password change, the user is required to change it immediately after logging in for the first time.

The new account is usable and is maintained according to the default security parameters unless you have set specific values for the user.

Modifying account creation defaults

For most users, the defaults should prove sufficient. If you selected to "Modify Defaults," you see the following form:



```
                                                                    Create

 Use the system default login group

 /                                                   Saturday August 31, 1990 1:06

 ------------------------- Make a new user account --------------------------
   ---------------------- New user account parameters ---------------------

       Login group    : Specify Default of
                        Value, (F3) for list :
       Groups         : [...           ]

       Login shell    : Specify [Default] of sh
                        Value, (F3) for list :

       Home directory : Specify [Default] of [ /usr/name                 ]
                        Value, (F3) for list : [                            ]
                        [ Create home ]   Do not create    Populate existing

       User ID number : [Specify] [Default] of [     ] value : [     ]

       Type of user   : Specify [Default] of individual
                        Value, (F3) for list :
       Account that may su(C) to this user   :
```

The cursor is initially positioned on the "Login group" field. Some of the fields displayed can only be modified at creation time, in the Modify mode. These fields are informational only and their values cannot be changed. The fields are as follows:

Login group
: This is the group associated with this account when the user logs in. This field can be changed, but must not be empty. It becomes the group field for this user in */etc/passwd*. Pressing function key ⟨F3⟩ provides a point-and-pick list of all currently existing groups.

Groups
: These are the groups this user is a member of. If you enter the name of a group that does not exist, it is automatically created as described in "Adding or changing groups."

Login shell
: This is the shell the user will use. (The default is defined in */etc/default/authsh*.) If a full pathname is supplied (as in */bin/sh*), the shell described by that pathname is simply used as the user's login shell. However, if the shell specified is not a pathname (as in *sh*), it is assumed to be the name of a "predefined shell", a shell defined in a subdirectory of */usr/lib/mkuser*. Choosing a predefined shell causes appropriate shell-related files (for example, *.profile* for *sh*) to be copied into the user's home directory when the account is created. (For information on creating your own predefined shells, see "Adding login shells and configuration files" later in this chapter.)

Home directory
: This defines where the user's files will reside. The default directory option is highlighted. Press ⟨Return⟩ to get the default location. The home directory options are as follows:

Create	creates a new directory for this user
Do not create	does not create a directory for this user
Populate existing	uses the existing directory specified

If you wish to have users share a single home directory, see the next section "Sharing home directories," for instructions.

User ID number
: This is the user identification (ID) number. Once set, a user's identity should not be changed as this would violate accountability.

Type of user

By default this is "individual," and you need not change it. Pseudo-users are anonymous accounts like *sysinfo* and *uucp*. Each pseudo-user has an "accountable user," who is considered responsible for that account. (For example, *root*, an individual, is defined as the accountable user for all pseudo-user accounts.) Several user types are provided for customers who need to make distinctions between accounts that can be used by different people.

Account that may su(C) to this user

This is the user responsible for this account. This field is only useful with the High security defaults, where strict accountability is maintained and use of **su**(C) is restricted. Under other defaults, this field can be ignored. This field can be changed if and only if the user is not an individual. For individual users this field is empty, but for non-individuals it must not be empty. It must contain the user name of another account. For example, all pseudo-user accounts shipped with the system are owned by *root*, and defined as an "individual" account. This can be set up so that every account can be traced to a real person. Press ⟨F3⟩ for a point-and-pick list of all users on the system.

Sharing home directories

You can set up user accounts to share the same login directory. To do this, create the directory normally during the adding of a new user. You should then exit **sysadmsh** and enter the following commands (replacing ***homedir*** with the real directory name):

```
cd homedir
chmod 775 .
chown auth .
```

In addition, enter one of the following commands according to the login shell used for the account:

Bourne or Korn shell

```
chmod 660 .profile
chmod 660 .kshrc (Korn shell only)
```

C-shell

```
chmod 660 .login .cshrc
```

This ensures that the members of the same login group can share this directory. Note that if you assign a different login group to a user, that user cannot share the directory. To add users that share the directory, be sure to select "Modify defaults," specify the directory, and "Populate existing" directory when creating subsequent users. Answer **n** when asked to overwrite the .files such as *.profile* and so forth.

Adding administrative users

In addition to the standard Identity information, users who act as administrators for printers, accounts, and so forth, can be assigned the responsibility by selecting Privileges. Under lower security defaults, most subsystem authorizations are already assigned by default, so this distinction is meaningless. (The **auth** authorization gives *root* powers and is only assigned to *root* by default, regardless of the security defaults.) Subsystems are discussed in "Changing default authorizations" later in this chapter, and assigning authorizations is discussed in the next section.

Altering the defaults displayed for user accounts

You can alter the default selections that appear in the user creation menu by editing the file */etc/default/authsh*. The following defaults can be redefined:

- login group
- groups
- login shell
- home directory
- range of user IDs
- type of user

The fields in */etc/default/authsh* are explained in the **authsh**(ADM) manual page.

Removing or retiring a user account

When operating under High (C2) security defaults, a user is never removed from the system. Once assigned, a user ID (UID) is never reused. Instead, a user account is "retired," or removed from service. Under other defaults, a user can be completely removed from the system. To retire a user account, make the following **sysadmsh** selection:

Accounts ⇨ User ⇨ Retire

When running with lower security defaults, you can completely remove accounts using the **rmuser**(ADM) utility. To remove a user, enter the following command, substituting the name of the user account for ***username***:

rmuser ***username***

Retiring or removing a user account does not remove the user's files; the system administrator must do this manually.

Unretiring a user account

If you are running under lower security defaults and you wish to "unretire" or reactivate a user's account, you can do so with the **unretire**(ADM) command. To reactivate a retired user, enter the following command, substituting the name of the user account for ***username***:

unretire ***username***

Locking or unlocking a user account

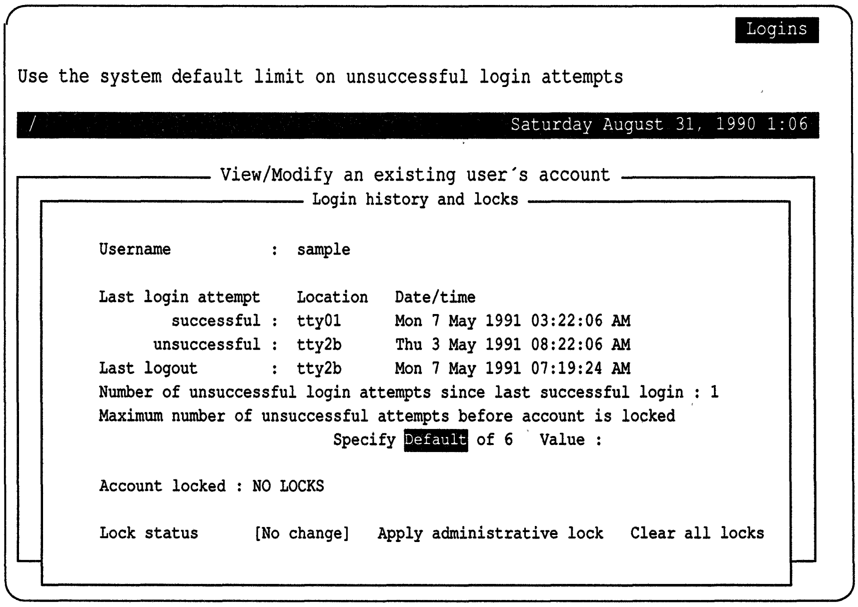
The system administrator can lock an account to prevent its use. In addition, under the Improved and High security defaults, an account is locked automatically if certain login parameters have been exceeded (see "Default account configuration" later in this chapter).

NOTE Under the Low and Traditional security defaults, accounts and terminals are only locked by the administrator; there are almost no limits on the number of unsuccessful login attempts, and password expiration is less strict.

Once a user or terminal is locked, only an administrator can unlock the user account or terminal (terminal locks are discussed in the next section). To lock or unlock an account, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Logins

The colon indicates that you must fill in a field (in this case the user name) before choosing the Logins selection. A form similar to the following is displayed:



```
                                                              Logins
Use the system default limit on unsuccessful login attempts

/                                          Saturday August 31, 1990 1:06

               ——— View/Modify an existing user's account ———
                     ——— Login history and locks ———

    Username          :  sample

    Last login attempt   Location   Date/time
            successful :  tty01      Mon 7 May 1991 03:22:06 AM
          unsuccessful :  tty2b      Thu 3 May 1991 08:22:06 AM
    Last logout         :  tty2b      Mon 7 May 1991 07:19:24 AM
    Number of unsuccessful login attempts since last successful login : 1
    Maximum number of unsuccessful attempts before account is locked
                             Specify Default of 6   Value :

    Account locked : NO LOCKS

    Lock status      [No change]   Apply administrative lock   Clear all locks
```

Move down to the "Lock status" field and toggle it to "Apply administrative lock" or "Clear all locks" as desired.

Locking or unlocking a terminal

To lock and unlock a terminal, respectively, use the following **sysadmsh** selections:

Accounts ⇨ Terminals ⇨ Lock
Accounts ⇨ Terminals ⇨ Unlock

When the prompt appears for the terminal, enter the name, for example: tty01. When a terminal is locked, the following message is displayed when an attempt is made to log in:

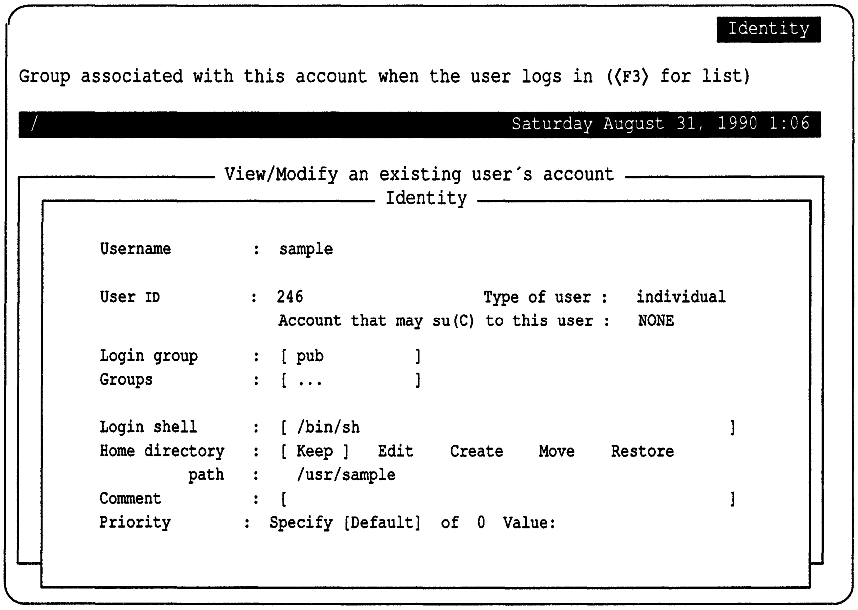
```
Terminal is disabled -- see Authentication Administrator
```

Changing a user's login group

To change a user's login group, make the following **sysadmsh** selection:

Accounts ⇨ User ⇨ Examine:Identity

The colon indicates that you must fill in a field (in this case the user name) before making the Identity selection. A form similar to the following is displayed:



```
                                                              Identity
Group associated with this account when the user logs in (<F3> for list)
/                                          Saturday August 31, 1990 1:06

                 View/Modify an existing user's account
                               Identity

     Username        :  sample

     User ID         :  246              Type of user :  individual
                        Account that may su(C) to this user :  NONE

     Login group     :  [ pub        ]
     Groups          :  [ ...        ]

     Login shell     :  [ /bin/sh                              ]
     Home directory  :  [ Keep ]  Edit   Create   Move   Restore
             path    :    /usr/sample
     Comment         :  [                                      ]
     Priority        :  Specify [Default]  of  0  Value:
```

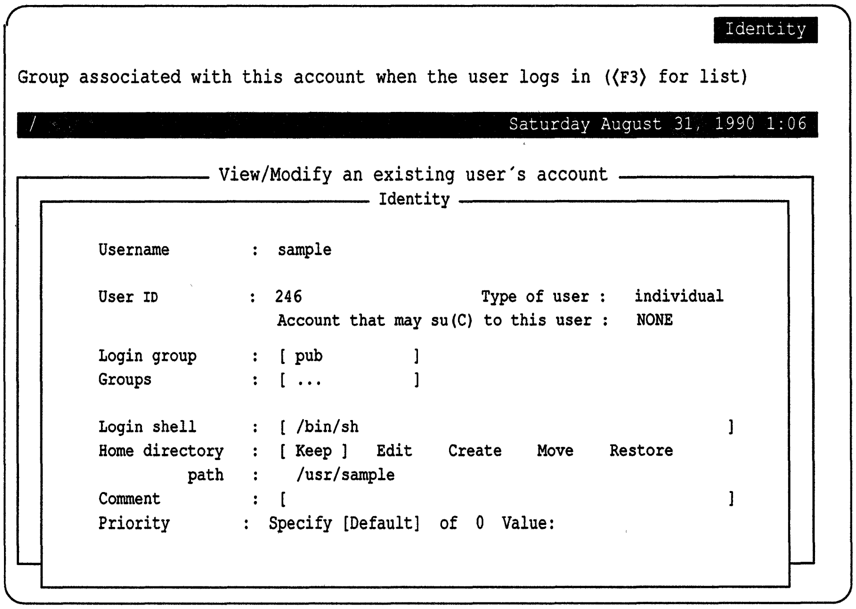
Modify the "Login group" field as desired.

Changing a home directory

To change a user's home directory, including moving the user's files, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Identity

The colon indicates that you must fill in a field (in this case the user name) before making the Identity selection. A form similar to the following is displayed:



```
                                                                  Identity

Group associated with this account when the user logs in (<F3> for list)

 /                                                   Saturday August 31, 1990 1:06

                View/Modify an existing user's account
                                  Identity

     Username          :  sample

     User ID           :  246                  Type of user :  individual
                          Account that may su(C) to this user :  NONE

     Login group       :  [ pub         ]
     Groups            :  [ ...         ]

     Login shell       :  [ /bin/sh                                      ]
     Home directory    :  [ Keep ]   Edit    Create    Move    Restore
               path    :    /usr/sample
     Comment           :  [                                              ]
     Priority          : Specify [Default]  of  0  Value:
```

After moving down to the "Home directory" field, you can select to do the following:

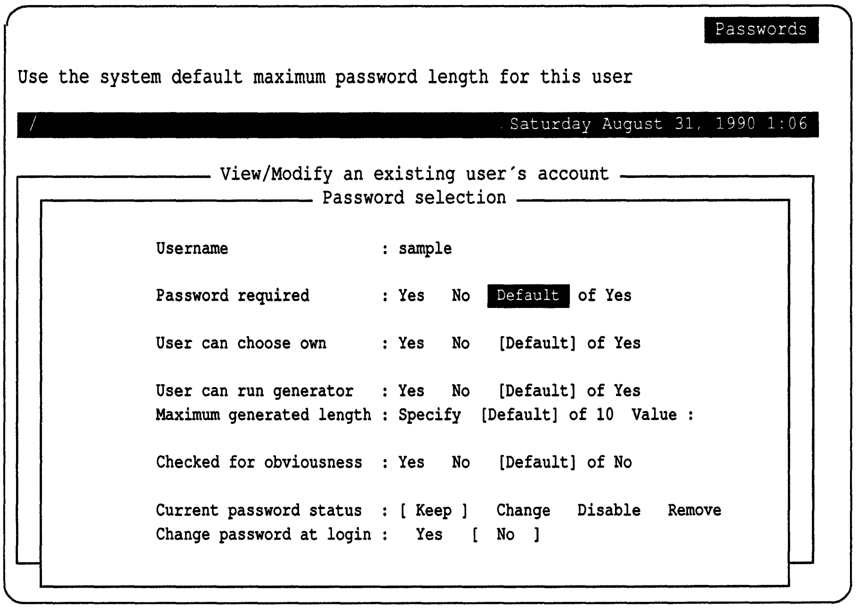
Keep	makes no change to the main directory for this user
Edit	changes this user's main directory path, but does not move any files
Create	creates the main directory for this user
Move	renames this user's main directory, moving all files in the old to the new
Restore	changes the path back to its previous value (no files are moved)

Changing a user password or password parameters

An administrator can change a user's password at any time. Password generation parameters can also be changed on an individual basis, just as they can be system-wide. This governs how a user's password is changed. To do this, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Password

The colon indicates that you must fill in a field (in this case the user name) before making the Password selection. The following form is displayed:



```
                                                            Passwords
Use the system default maximum password length for this user

 /                                          Saturday August 31, 1990 1:06

                  View/Modify an existing user's account
                         Password selection

        Username                 : sample

        Password required        : Yes   No   Default  of Yes

        User can choose own      : Yes   No   [Default] of Yes

        User can run generator   : Yes   No   [Default] of Yes
        Maximum generated length : Specify  [Default] of 10  Value :

        Checked for obviousness  : Yes   No   [Default] of No

        Current password status  : [ Keep ]   Change   Disable   Remove
        Change password at login :   Yes    [  No  ]
```

The Password selection parameters are similar to the system-wide parameters described in "Changing default password restrictions" later in this chapter. The following parameters define the per-user password restrictions:

Password required

If Yes, the user cannot log in without a password; if No, the user can log in without a password.

User can choose own

This parameter determines whether or not users can choose their own passwords. If this parameter is set to Yes, users can pick their passwords. If that parameter is set to No, the system must generate a password for that user, according to the random password generation procedures.

NOTE If "User can choose own" and "User can run generator" are both No, users cannot change their own passwords.

User can run generator
: This parameter enables the user to run the password generator. Note that this does not allow the user to choose a password, but merely to generate a new random password.

Maximum generated password length
: This parameter is the maximum length of a password generated by the system for this user. The maximum is 80 characters.

Checked for obviousness
: This parameter is only valid with Improved or High security defaults, or when **GOODPW=YES** appears in */etc/default/passwd*. This controls whether the system should run simple or complex triviality checks on a new password. Triviality checks increase the time required to change a password substantially (see "Customizing password checking with goodpw(ADM)" later in this chapter).

Current password status
: You can choose one of the following:

Keep	does not change password.
Change	invokes the password change procedure described at the end of "Adding a user"
Disable	disables the password, which effectively locks the user out
Remove	removes the password, enabling the user to log in without a password

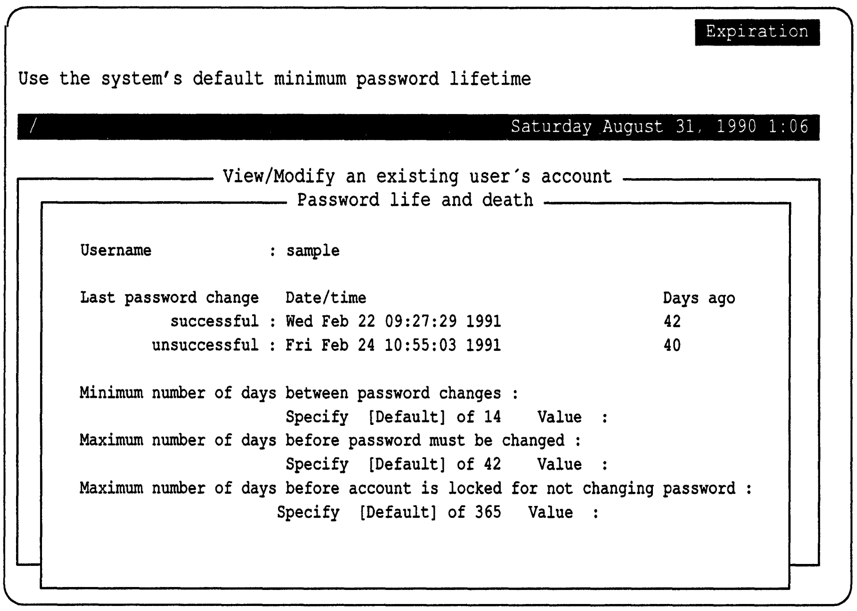
Change password at login
: This option allows you to force users to change their password the next time they log in.

Altering user password expiration parameters

It is sometimes useful to define expiration parameters for a user that differ from the system defaults. To do this, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Expiration

The colon indicates that you must fill in a field (in this case the user name) before choosing the Expiration selection. A form similar to the following is displayed:



```
                                                                Expiration
Use the system's default minimum password lifetime

/                                                 Saturday August 31, 1990 1:06

                  View/Modify an existing user's account
                          Password life and death

    Username           : sample

    Last password change   Date/time                              Days ago
            successful : Wed Feb 22 09:27:29 1991                 42
          unsuccessful : Fri Feb 24 10:55:03 1991                 40

    Minimum number of days between password changes :
                           Specify  [Default] of 14    Value  :
    Maximum number of days before password must be changed :
                           Specify  [Default] of 42    Value  :
    Maximum number of days before account is locked for not changing password :
                          Specify  [Default] of 365   Value  :
```

The user parameter descriptions are similar to the system-wide parameters described in "Changing default password restrictions" later in this chapter. The descriptions differ, but the parameters are the same. They control the time that a password is valid, and the procedures for changing the password once it becomes invalid. The lifetime of a password is divided into three intervals:

- The password is valid.
- The password has expired; the user can still log in and change it (if authorized to do so).
- The password is dead; the user is locked out and the administrator must unlock the account and the user change the password.

NOTE The Low and Traditional security defaults use password restrictions that are very lenient; passwords do not expire, accounts are not locked, and there is no minimum interval between password changes.

The system also stores a minimum time between password changes. This prevents users from changing their passwords when they expire and then immediately changing them back to one they remember. A user's password cannot be changed until the minimum time has been exceeded. The password lifetime scheme is implemented as follows:

Minimum number of days between password changes
: This defines the number of days users must wait before they can change their password.

Maximum number of days before password must be changed
: This defines the length of time a given password is valid.

Maximum number of days before account is locked for not...
: This defines the interval between the last password change and when the password dies.

The default account initialization files (*.cshrc*, *.profile*, *.kshrc*, and so forth) automatically execute the **prwarn**(C) utility at login time to warn users about impending password expiration.

Changing user authorizations

Subsystem authorizations are discussed later in this chapter in "Changing default authorizations." Additional information is located in the the "Maintaining system security" chapter of this guide. Authorizations define what users are allowed to do.

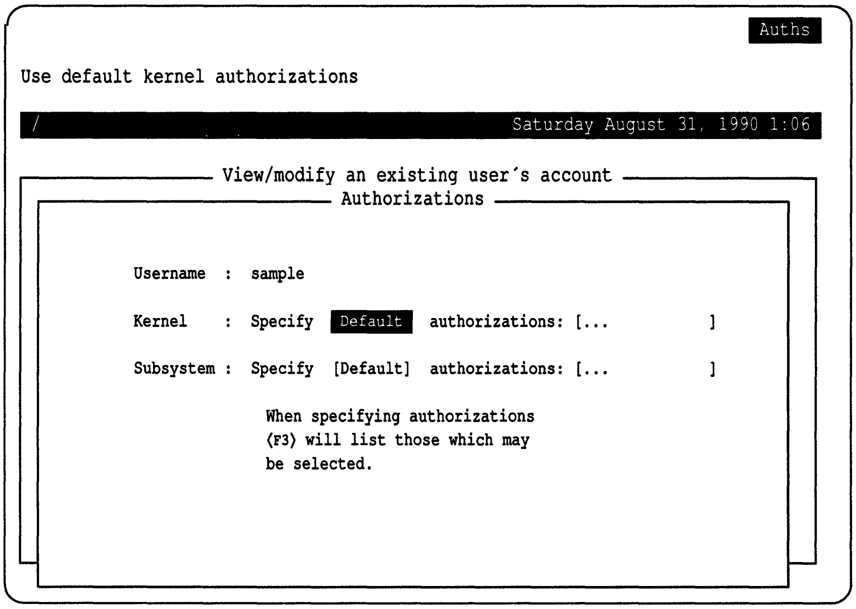
NOTE The Low and Traditional security defaults assign most authorizations to users by default and it should not be necessary to change them. With the High (C2) security defaults, few authorizations are assigned by default; they are intended for users entrusted with administration of a subsystem.

To assign a new authorization to a user, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Privileges

The colon indicates that you must fill in a field (in this case the user name) before choosing the Privileges selection.

The following form is displayed:



```
                                                                    Auths

Use default kernel authorizations

 /                                              Saturday August 31, 1990 1:06

 ─────────────── View/modify an existing user's account ──────────────────
   ─────────────────────── Authorizations ──────────────────────────


         Username  :  sample

         Kernel    :  Specify  Default   authorizations: [...          ]

         Subsystem :  Specify  [Default]  authorizations: [...          ]

                        When specifying authorizations
                        <F3> will list those which may
                        be selected.
```

You can select Specify and press ⟨F3⟩ to open a window that lists the available authorizations. Use the arrow keys to move up and down the list. You can select multiple authorizations by pressing the space bar; each item selected is marked with an asterisk (*).

NOTE If you switch from defaults to specified, the default values are eliminated for that user; only those authorizations you specify are in effect.

Changing user audit parameters

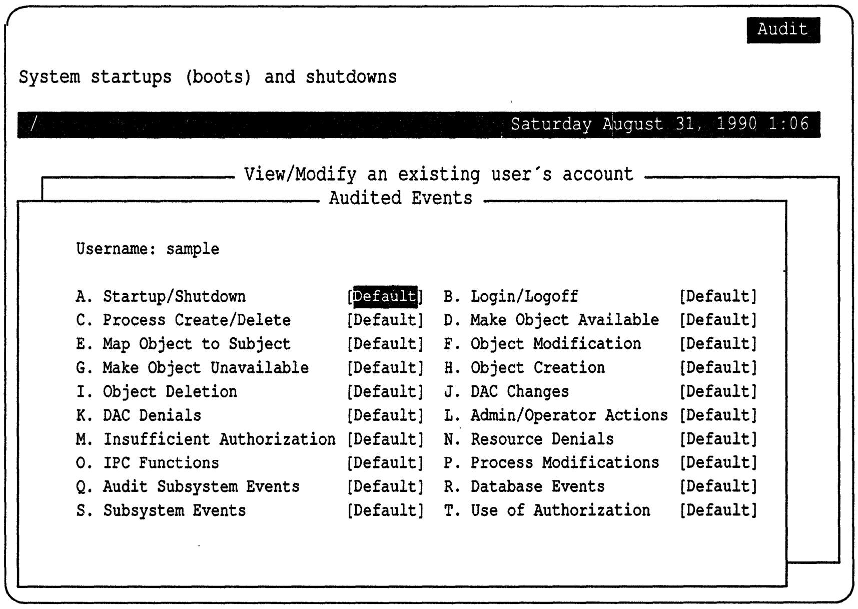
You can define audit parameters for individual users just as with the system-wide parameters. Any settings defined for a user override the system defaults.

NOTE It is not necessary to modify audit parameters if you do not intend to use auditing.

To define or change audit parameters, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Audit

The colon indicates that you must fill in a field (in this case the user name) before choosing the Audit selection. A form similar to the following is displayed:



```
                                                                    Audit

System startups (boots) and shutdowns

 /                                           Saturday August 31, 1990 1:06

                 View/Modify an existing user's account
                          Audited Events

    Username: sample

    A. Startup/Shutdown             [Default]  B. Login/Logoff             [Default]
    C. Process Create/Delete        [Default]  D. Make Object Available    [Default]
    E. Map Object to Subject        [Default]  F. Object Modification      [Default]
    G. Make Object Unavailable      [Default]  H. Object Creation          [Default]
    I. Object Deletion              [Default]  J. DAC Changes              [Default]
    K. DAC Denials                  [Default]  L. Admin/Operator Actions   [Default]
    M. Insufficient Authorization   [Default]  N. Resource Denials         [Default]
    O. IPC Functions                [Default]  P. Process Modifications    [Default]
    Q. Audit Subsystem Events       [Default]  R. Database Events          [Default]
    S. Subsystem Events             [Default]  T. Use of Authorization     [Default]
```

A detailed description of audit events is found in the "Using the audit subsystem" chapter of this guide.

There are three possible settings for each event:

Default	use the system-wide account defaults
Always	always audit this event (overrides system-wide default)
Never	never audit this event (overrides system-wide default)

You can press ⟨F3⟩ to select from a list of these settings or fill them in manually. Abbreviations are recognized (for example, "n," "nev," and "N" all mean Never). To execute the form, press ⟨Ctrl⟩x. (If you fill in the last field on a form, it is automatically executed.)

Adding or changing groups

To add a group, simply enter a new group name while creating or altering a user account. You are prompted that the group does not exist and asked to confirm that you wish to create the new group.

You can also edit the file */etc/group* and add or modify the contents as desired.

WARNING Do not change the GIDs of any default system groups.

Supplemental groups

On most System V implementations, a user can be a member of several groups, but can only have a single effective group ID (GID), which limits membership to one group at a time. The effective GID (and group membership) is changed using the **newgrp**(C) command. The list of groups is located in */etc/group*.

The supplemental groups feature allows an additional set of groups to be defined, permitting simultaneous membership in multiple groups. This provides additional access permissions on top of those provided by the user and group IDs.

The *.suppgroups* file, which is recognized by programs that establish identity (**login**, **su** and **cron**), specifies which groups (of which the user is a member) are placed in the user's supplemental group list when identity is established. The *.suppgroups* file contains one group name per line. If a *.suppgroups* file does not exist, the initial supplemental group list is filled with the login group ID followed by the groups the user is listed as being a member of in */etc/group*. They are listed in the same order as in */etc/group*. The *.suppgroups* file does not need to be readable or writable by the user.

The **sg**(C) command is used to manipulate a user's supplemental group set. This command does not establish identity so it does not read the *.suppgroups* file.

Changing the maximum number of supplemental groups

By default, the maximum size of a supplemental group set is eight, meaning that a user can only be in eight groups at one time. This number is controlled by the **NGROUPS** tunable kernel parameter. This value can be changed by invoking the **sysadmsh** selection

System ➪ Configure ➪ Kernel ➪ Parameters

and selecting category 3, "Files, Inodes, and Filesystems," and changing the value of **NGROUPS**. The kernel must then be relinked and booted for the new value to take effect. Use the **sysadmsh**

System ➪ Configure ➪ Kernel ➪ Rebuild

selection to relink the kernel. See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for complete instructions.

Allowing users to execute super user commands

It is possible to assign users the capability of executing *root*-only commands without giving them complete *root* powers. This is done using the **asroot**(ADM) utility to create a new authorization associated with the command you wish to assign. You can then add this authorization to any user. The procedure for setting up a command is described in detail in the **asroot**(ADM) manual page.

NOTE The **asroot** utility requires the user's password to be re-entered when the system is configured with the High security defaults.

Accessing other accounts with su(C)

The **su**(C) utility (for super user) can be used to switch over to another account temporarily. It is primarily used to access the *root* account, when it is executed without an argument. Otherwise, it is used in the following form:

su *username*

su prompts for the account password, and if it is correct, a Bourne shell is started under the other account. Transitions with **su** do not affect the login user ID (LUID), so login and audit records remain accurate. If a dash "**-**" is included in the command (**su -**), the environment for that user is executed (including login shell, home directory, and so forth). To exit the shell, enter **exit** or press ⟨Ctrl⟩**d** and you are returned to your own account.

Users can **su** to an account they are defined as responsible for in the "Account that may su(C) to this user" field of the Accounts ➪ User ➪ Examine:Identity **sysadmsh** selection. To access the *root* account (or any other account they are **not** responsible for), however, the user must have the **su** authorization. This can be assigned using the Accounts ➪ User ➪ Examine:Privileges selection (see "Changing user authorizations" earlier in this chapter).

NOTE The Low, Traditional, and Improved security defaults assign the **su** authorization by default and users can **su** to any account if they know the password. Under the High security defaults, the **su** authorization is not assigned.

Moving user accounts to another filesystem or directory

If you want to move your user accounts to a different location (such as a new filesystem or directory), you must start by ensuring that new accounts are placed in the new location. Here we use a hypothetical filesystem called */x* as an example. The **sysadmsh**(ADM) Accounts option (used to create new user accounts) reads the default location for user accounts from the */etc/default/authsh* and */usr/lib/mkuser/homepaths* files.

Edit the files */etc/default/authsh* and */usr/lib/mkuser/homepaths*. Change the entries that read "/usr" to "/x". This establishes */x* as the location for new user accounts.

Whenever you run the **sysadmsh** Accounts selection to add a new user, that user account is placed in */x*. (Make certain that the filesystem containing the user accounts is mounted before you create a new user, or the user's home directory will not be accessible when the filesystem is mounted.)

If there are existing user accounts that you wish to relocate, you should first make a backup floppy or tape of all home directories you plan to move. (This protects your users from possible loss of data.) Next, change the home directory for each user by calling up their account information with the following **sysadmsh**(ADM) selection:

Accounts ➪ User ➪ Examine:Identity

Move to the "Home Directory" field and select the Move option. Follow the prompts. Do this for every user whose home directory has changed.

Migrating user accounts to non-SCO-based UNIX systems

You can migrate user accounts to other XENIX or UNIX systems by copying the */etc/passwd* (and */etc/shadow*, if applicable) to the target system. If the target system does not use */etc/shadow*, you should run the **pwunconv** utility to consolidate the information into */etc/passwd* before copying the file to the target system.

NOTE If you need to migrate accounts to another SCO-based UNIX system, use the **ap**(ADM) utility described in the next section.

Password compatibility

The password encryption scheme used by the system maintains compatibility with other XENIX and UNIX system implementations, while providing the ability to create passwords with more than eight significant characters. However, if you are using the Improved or High security defaults, passwords of up to 80 characters are allowed, which cannot be imported to other systems. The **sysadmsh** selection Accounts ➪ Defaults ➪ System includes the field "Length of cleartext password". A value of 1 (instead of 10 as used in the Improved or High defaults) allows the encryption mechanism to ignore characters following the first eight, thus allowing complete compatibility with other systems. See "Configuring database recovery and precedence" earlier in this chapter for more information.

Migrating user accounts to SCO-based UNIX systems

You can easily duplicate user accounts on other SCO-based UNIX systems (as with a network) using the **ap**(ADM) utility. This utility creates a profile containing all account data for one or more users.

WARNING The **ap**(ADM) utility does not create profiles that are portable to non-SCO-based UNIX systems. If you need to migrate accounts from a non-SCO-based UNIX system or SCO XENIX system, use the **addxusers**(ADM) utility described in the next section.

Account information is gathered from the */etc/passwd* file and the Protected Password database. Irrelevant information about the user (including unsuccessful logins, unsuccessful password changes, and the location and time of last login) is not included in the profile.

To create a profile and install it on another machine, do the following:

1. Log in as *root* and enter the following command on the machine where the accounts reside:

   ```
   ap -d -v usernames > profile.acct
   ```

   usernames is the list of one or more user names.

2. Log in as *root* and move the *profile.acct* file to the target machine (use **tar**, or **cp** if your machine is on a network).
3. Enter the following command:

   ap -r -f profile.acct ***usernames***

The new account information is in place and ready for use.

Migrating user accounts from SCO XENIX or non-SCO-based UNIX systems

To copy accounts from other non-SCO-based UNIX systems (or SCO XENIX systems) to your system, use the **addxusers**(ADM) utility. **addxusers** accepts an edited */etc/passwd* file as input and makes the necessary database modifications for use on your system. Refer to the **addxusers**(ADM) manual page for more information.

WARNING Most XENIX and UNIX systems only use the first eight characters for encryption. This can cause unexpected results when moving an encrypted password string from one of these systems to an SCO-based UNIX system. If a password longer than eight characters has been used, such as "narcissus", only the first eight characters ("narcissu") should be entered on an SCO-based UNIX system.

Default account configuration

This section explains how to alter the system security defaults, which include default password schemes, subsystem authorizations, and number of login attempts permitted for users.

The system includes four sets of defaults that define the security scheme for user accounts:

- Low
- Traditional
- Improved
- High

The "High" defaults are designed to meet the requirements set forth by the Department of Defense's *Trusted Computer System Evaluation Criteria* (also known as TCSEC or the *Orange Book*). Table 4.1 lists the defaults used by each set. The Low and Traditional defaults use values consistent with non-trusted UNIX systems. Selecting the security defaults is described later in this section.

NOTE The pre-defined defaults can be customized as desired; the four default sets are provided for ease of configurability.

Table 4-1 System default security parameters

	Security level			
Security parameters	Low	Traditional	Improved	High
Passwords				
Minimum days between changes	0	0	0	14
Expiration time (days)	infinite	infinite	42	42
Lifetime (days)	infinite	infinite	365	90
User can choose own	yes	yes	yes	no
User can run generator	yes	yes	yes	yes
Maximum generated length	8	8	10	10
Minimum length	1	3	5	8
Checked for obviousness	no	UNIX	yes[1]	yes[2]
Password required to login	no	no	yes	yes
Single user password required	yes	yes	yes	yes
Logins				
Maximum unsuccessful attempts (account/terminal)	infinite	99	5/9	3/5
Delay between login attempts (secs)	0	1	2	2
Time to complete login (secs)	60	60	60	60
Authorizations				
Subsystem	audittrail, backup, lp, mem, queryspace, shutdown, su, terminal	audittrail, mem, printqueue, queryspace, terminal, su	audittrail, queryspace, printqueue, su	queryspace,
Kernel	chmodsugid, chown, execsuid, suspendaudit	chmodsugid, chown, execsuid	chmodsugid, chown, execsuid	chown, execsuid
Default umask[3]	022	022	027	077
C2 Features				
LUID enforcement[4]	no	no	no	yes
STOPIO on devices[4]	no	yes	yes	yes
SUID/SGID clear on write[4]	no	yes	yes	yes
Users can be deleted[5]	yes	yes	yes	no
Database corruption[6]	recover	recover	lockout	lockout
Database precedence[7]	UNIX	UNIX	TCB	TCB
Other				
Users can schedule jobs	allow	allow	deny	deny
Home directory mode	755	755	750	700
Dialup printers allowed	yes	yes	no	no
Hushlogin allowed[8]	yes	yes	yes	no
Password for asroot(ADM)	no	no	no	yes
Significant characters in passwords	8	8	80	80
su(C) use logged	no	yes	yes	yes
/etc/shadow present	no	yes	yes	yes

Notes:

1. Simple checks are made, such as disallowing user names, machine names, etc...
2. Thorough checks are made, including checking that words are correctly spelt.
3. These are located in */etc/profile* and */etc/cshrc*. A **umask** of 077 results in the creation of files that are readable only by the owner.
4. These features are explained in the "Maintaining system security" chapter.
5. A requirement central to C2 is that a user ID (UID) cannot be reused. This means that user accounts cannot be reused or reactivated after retirement. With the lower security defaults, user accounts can be removed rather than retired and user IDs can be altered or reused.
6. On a system that conforms to C2 requirements, users are locked out of a system when a security database becomes corrupted. This ensures that the system does not operate in a potentially non-secure state. In the lower defaults, the system attempts to correct inconsistencies automatically and displays a warning rather than locking out users.
7. Two sets of account databases are maintained: UNIX System V and trusted computing base (TCB) files. One set is used as a master when a discrepancy occurs. This is described in "Configuring database recovery and precedence" in this chapter.
8. This feature allows the suppression of login messages. See **login**(M) for information.

Changing the security defaults

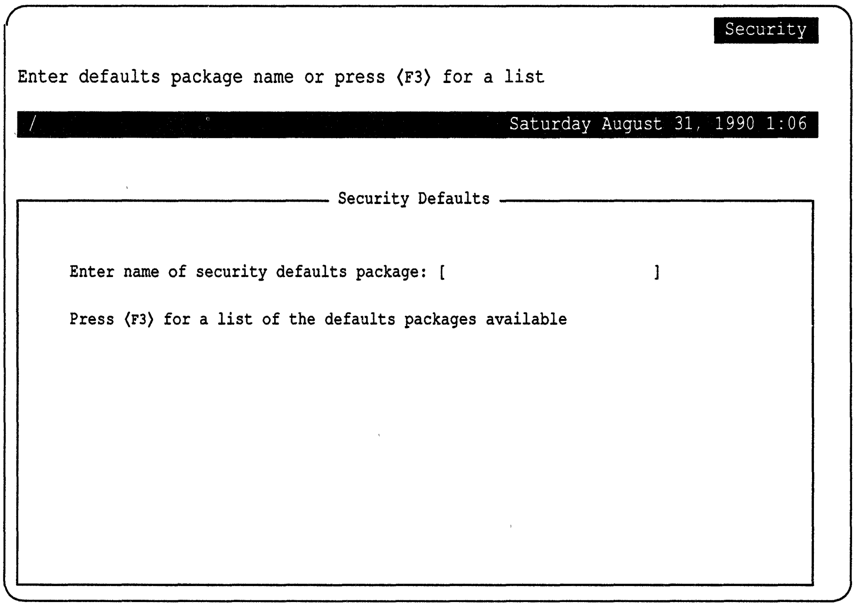
You were given the choice of security defaults at installation time. It is possible to later select another set of defaults. Should you wish to change the security defaults of your system, make the following **sysadmsh** selection:

System ⇨ Configure ⇨ Security

This allows you to select an alternate set of defaults (see Table 4.1).

NOTE After having selected lower security defaults, it is possible to select the Improved or High defaults, although this does not mean that your system automatically conforms to the requirements of a C2 system. (By definition, a C2 system must adhere to the requirements from initial installation.)

When you make the Security selection, the following is displayed:



```
                                                                Security

Enter defaults package name or press <F3> for a list

 /                                              Saturday August 31, 1990 1:06

                             Security Defaults

    Enter name of security defaults package: [                        ]

    Press <F3> for a list of the defaults packages available
```

You may reconfigure your system security to suit your own requirements. Four levels of preconfigured security defaults are available:

High	recommended for systems containing confidential information and accessed by many users. Passwords are strictly controlled and assigned to users; they cannot choose their own. User accounts cannot be removed or reactivated. All C2 features are engaged and database corruption results in a lockout of all users until the administrator fixes the problem.
Improved	recommended for systems accessed by groups of users who can share information. Password expiration is more lenient and users can choose their own passwords. LUIDs are not enforced, and user accounts can be removed or reactivated as desired. Database corruption results in system lockout.
Traditional	provided for compatibility with existing UNIX systems. Passwords do not expire and standard System V password checking is used. Passwords are not required. Database corruption is handled transparently.
Low	recommended only for systems not publicly accessible with a small number of cooperating users. No C2 features are engaged and no password checking is done. The */etc/shadow* does not exist by default.

The system is designed to meet the requirements for the C2 level of trust describing the protection given to prevent unauthorized access to a system and its data. The High and Improved levels are suitable for C2 systems.

Changing system account parameters dynamically

In addition to selecting a package of default parameters, the following system-wide account parameters can be customized individually:

- authorizations
- password
- logins

The system-wide security parameters control the way that users log in and, once they establish a session, the terminal and authorization environment that the system presents to them. Each parameter that you can change from the **sysadmsh** interface is discussed here. Other parameters that affect system operation are addressed later.

You should use the system-wide functions to define your own default system behavior. Then use the user-specific functions to adjust that behavior for any user having different requirements. As you might expect, the user-specific entries override the system defaults for any given user.

Changing default login restrictions

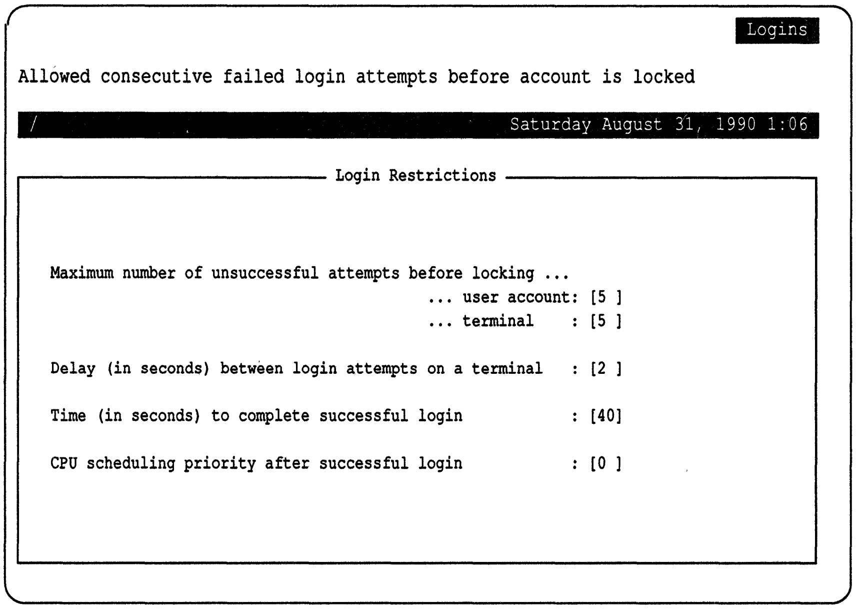
Most of the configurable security parameters deal with the way the system creates a login session. These include login particulars and the way that passwords are generated and enforced. The login parameters enforce the account and terminal-locking features. When users log in, they must give a login name and password. In addition, the user has a limited number of tries to log in. There is a limit on the number of times an unsuccessful login attempt can occur before either the account or the terminal are locked. If either count is exceeded, the user or the terminal is locked against future login. This feature guards against penetration attempts by restricting the number of times a malicious user (or computer programmed by a malicious user) can try to break into the system.

NOTE The Low and Traditional security defaults use login restrictions that are very lenient; there are almost no limits on the number of unsuccessful login attempts and little delay between attempts.

To access the login restriction parameters, make the following **sysadmsh** selection:

Accounts ➪ Defaults ➪ Logins

The following form is displayed:



```
                                                                      Logins
Allowed consecutive failed login attempts before account is locked

 /                                               Saturday August 31, 1990 1:06

                             Login Restrictions


  Maximum number of unsuccessful attempts before locking ...
                                        ... user account: [5 ]
                                        ... terminal    : [5 ]

  Delay (in seconds) between login attempts on a terminal  : [2 ]

  Time (in seconds) to complete successful login          : [40]

  CPU scheduling priority after successful login          : [0 ]
```

The parameters are described as follows:

Maximum number of unsuccessful attempts before locking
: This is the system default number of unsuccessful attempts allowed for users and terminals. If a particular user or terminal needs either a more restrictive or more permissive number, the user's account can be modified or the terminal's configuration (see "Terminal login management" in the "Maintaining system security" chapter) can be changed to override the system default.

Delay (in seconds) between login attempts on a terminal
: This parameter controls the amount of time that must pass between unsuccessful login attempts. To further reduce the possibility of penetration, the system can delay between login attempts to increase the amount of time it takes to try to log into the system repeatedly. You can increase this parameter to control the cycle time of the "login:" prompt. By combining this parameter with the user and terminal unsuccessful attempt parameters, you can frustrate attempts to try passwords repeatedly on certain (or a combination of) terminal lines.

Time (in seconds) to complete successful login
: This parameter determines how much time users have to enter their name and password before the login attempt is terminated.

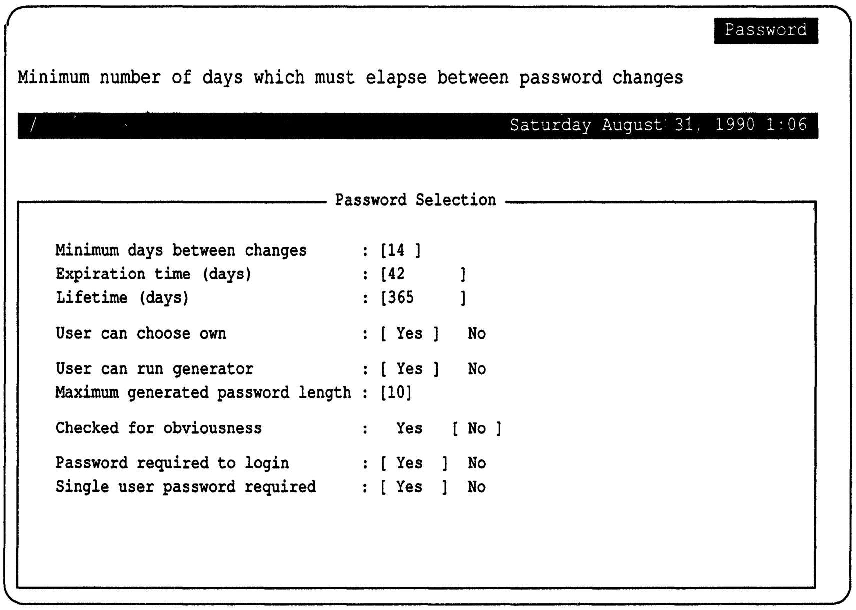
CPU scheduling priority after successful login
: This sets the **nice**(C) value associated with this user's processes. This allows you to set a higher or lower CPU priority for a user. See the **nice**(C) manual page for details.

Changing default password restrictions

Given that you can control the number of attempts an intruder can try to guess a password, the remaining task is to control the complexity of the password itself. To access the password restriction parameters, make the following **sysadmsh** selection:

Accounts ➪ Defaults ➪ Password

The following screen is displayed:



```
                                                                    Password

Minimum number of days which must elapse between password changes

 /                                                   Saturday August 31, 1990 1:06

                                 Password Selection

    Minimum days between changes      : [14 ]
    Expiration time (days)            : [42      ]
    Lifetime (days)                   : [365     ]

    User can choose own               : [ Yes ]   No

    User can run generator            : [ Yes ]   No
    Maximum generated password length : [10]

    Checked for obviousness           :   Yes   [ No ]

    Password required to login        : [ Yes  ]  No
    Single user password required     : [ Yes  ]  No
```

The types of password checking the system does is controlled by the parameters set on this screen. The parameters control the time that a password is valid, and the procedures for changing the password once it becomes invalid. A password is valid until it "expires" or "dies." An expired password can be changed by whomever is authorized to change passwords for the account. A password expires when its expiration time is reached. The expiration time can be set from **sysadmsh** on a system-wide or a per-user basis, and it is expressed in number of days from the time that the password was last changed. A dead password causes the user account to be locked. Only the administrator can unlock the user's account, which is then treated as an account with an expired password. The password must still be changed before the user can log in again.

NOTE The Low and Traditional defaults use password restrictions that are very lenient; passwords do not expire, accounts are not locked, and there is no minimum interval between password changes. In addition, there is no password checking with the Low defaults; Traditional defaults use the customary UNIX system password checks.

The system also stores a minimum time between password changes. This prevents users from changing their passwords when they expire and then immediately changing them back to one they remember. A user's password cannot be changed until the minimum time has been exceeded. This parameter may also be set on a per-user or system-wide basis.

The following parameters define the password restrictions:

Minimum days between password changes
: The number of days a user must wait between password changes.

Expiration time (days)
: This defines the length of time a given password is valid. The default account initialization files (*.cshrc, .profile, .kshrc,* and so forth) automatically execute the **prwarn**(C) utility at login time to warn users about impending password expiration.

Lifetime (days)
: This defines the interval between last password change and when the password dies.

User can choose own
: This parameter determines whether or not users can choose their own passwords. You can choose to have the system generate passwords automatically for users. This guards against users picking "obvious" passwords that a knowledgeable intruder could guess given some personal facts about the user. Most UNIX systems allow users to pick their passwords. If this parameter is set to Yes, then rules consistent with non-trusted UNIX systems are in effect, allowing users to pick their passwords. If that parameter is set to No, the system must generate passwords for that user, according to the random password generator.

User can run generator
: This parameter enables users to run the password generator. Note that this does not allow users to choose a password, merely generate a new random password.

Maximum generated password length
: This defines the maximum length of a password generated by the system. The maximum is 80 characters.

Checked for obviousness

This parameter is only valid with the Improved or High security defaults, or when **GOODPW=YES** appears in */etc/default/passwd.* This controls whether the system should run simple or complex triviality checks on passwords. These checks assure that the password does not appear in the online dictionary, along with the other checks described in **goodpw**(ADM.) Setting this parameter to Yes ensures the failure of some penetrations based on dictionary checking, but this can be controlled more effectively through the limits on account and terminal logins. Triviality checks increase the time required to change a password substantially. (See "Customizing password checking with goodpw(ADM)" later in this chapter).

Password required to login

If No, user accounts can be without passwords; if Yes, a password must exist for the account.

Single user password required

This governs whether a password is required to bring the system up in single-user (maintenance) mode.

When an account is locked by the system, only *root* or the accounts administrator can unlock it. The password must be changed at that time. You can override these parameters for any user as described earlier in "Changing a user password or password parameters."

Customizing password checking with goodpw(ADM)

The **goodpw**(ADM) utility allows you to customize password checking. The file */etc/default/goodpw* contains the password control settings. These settings allow you to specify if passwords are checked against dictionary words, word rotations, user, group, and system names.

NOTE Password checking can also be set by editing the */etc/default/passwd* and changing the value of **GOODPW=** as follows:

YES	use **goodpw**
NO	use standard UNIX system checking
NONE	perform no password checking

In addition, the directory */usr/lib/goodpw/checks* allows you to further customize password requirements according to the type of password:

- user
- filsys (filesystem)
- group
- modem

Each type has a file (*secure* or *weak*) that is used depending on whether the "Check for obviousness" field is set to Yes (secure) or No (weak).

You can also define regular expressions (character combinations and arrangements) that all passwords must match, or must not match, with the files */usr/lib/goodpw/match* and */usr/lib/goodpw/reject*, respectively. For more details, refer to the **goodpw**(ADM) manual page.

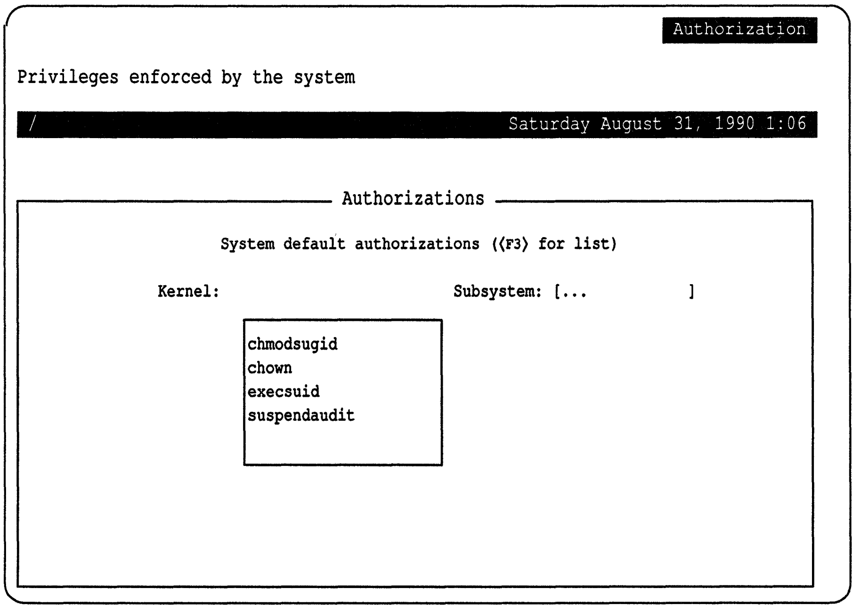
Changing default authorizations

There are two types of authorizations: *Subsystem authorizations* are associated with users and allow the user to execute trusted utilities. *Kernel authorizations* are associated with processes and allow a process to perform certain actions. Each user session has a set of kernel authorizations and subsystem authorizations.

To access the authorization parameters, make the following **sysadmsh** selection:

Accounts ➪ Defaults ➪ Authorizations

The following screen is displayed:



```
                                                      Authorization
Privileges enforced by the system

 /                                          Saturday August 31, 1990 1:06

                              Authorizations
                 System default authorizations (<F3> for list)

            Kernel:                         Subsystem: [...          ]

                       chmodsugid
                       chown
                       execsuid
                       suspendaudit
```

You can use the ⟨Tab⟩ key to move between kernel and subsystem authorizations. Use the ⟨F3⟩ key to get a pop-up window that lists each set of authorizations. Detailed descriptions of authorizations appear in the sections that follow.

Subsystem authorizations

Subsystem authorizations were designed to implement administrative roles rather than using a single *root* user to administer the system. Under the Low and traditional security defaults, almost all subsystem authorizations are assigned to users by default.

If you intend to operate a system that conforms to C2 requirements, you should grant subsystem authorizations based on the notion of *least-privilege,* where users are assigned subsystem authorizations based on their responsibilities. For example, the accounts administrator is given **auth** authorization and the printer administrator is given **lp** authorization. Under this scheme, general users should be assigned as few subsystem authorizations as possible. The subsystem authorizations are listed in Table 4.2.

Table 4-2 Subsystem authorizations

Authorization	Subsystem	Powers
mem	Memory	access to system data tables, listing all processes on the system
terminal	Terminal	unrestricted use of the **write**(C) command
lp	Line Printer	administer printers
backup	Backups	perform backups
auth	Accounts	administer accounts: adding users, changing passwords, and so on
audit	Audit	audit administrator: run system audits and generate reports
cron	Job Scheduling	control use of **cron**(C), **at**(C), and **batch**(C) commands
sysadmin	-	not implemented
passwd	Passwords	ability to change user passwords

Secondary authorizations allow limited access by users to resources that would otherwise be tightly controlled (for example, without the **printqueue** authorization, users would only be able to see their own jobs when they use the **lpstat** command). They are useful when running the Improved or High security defaults to provide behavior that is more consistent with other UNIX operating systems. See Table 4.3.

Table 4-3 Secondary authorizations

Secondary authorization	Subsystem	Description
audittrail	audit	ability to generate personal audit reports on one's own activities
queryspace	backup	use of **df**(C) command to query disk space
printqueue	lp	view all jobs in queue using **lpstat**(C)
printerstat	lp	use of printer enable/disable commands
su	auth	access to the *root* (super user) account and other accounts. (Access still requires password.)

NOTE When the primary authorization for a subsystem is granted, the secondary authorizations for that subsystem are also granted. (For example, the **lp** authorization carries the **printqueue** and **printerstat** authorizations.)

Kernel authorizations

The kernel authorizations govern the power that user processes have to execute specific operating system services. For example, the ability to change ownership of a file is governed by the **chown** authorization. (The **chown** authorization allows the use of the **chown**(S) system call which enables **chown**(C) to work.) The default kernel authorizations are used whenever a user's kernel authorizations are unspecified. Thus, users that need more authorization can have user-specific entries that grant them those authorizations, while normal users can have their authorizations set to the system-wide defaults. See Table 4.4.

NOTE Restricted **chown** is required for NIST FIPS 151-1 conformance. The **chown** authorization should not be assigned to users if you wish to conform to NIST FIPS 151-1 requirements.

Table 4-4 Kernel authorizations

Authorization	Action
configaudit	configure audit subsystem parameters
writeaudit	write audit records to the audit trail
execsuid	ability to run set-UID programs
chmodsugid	ability to set the set-UID and set-GID bit on files
chown	ability to change the owner of an object
suspendaudit	suspend operating system auditing of the process

Under the Low and Traditional security defaults, most kernel authorizations are assigned by default and should not require modification. Under the High security defaults, **chmodsugid** is not assigned by default. The **configaudit** and **writeaudit** parameters apply specifically to audit operations and should not be assigned to users; these parameters are explained in "Using the audit subsystem" chapter in this guide.

The **execsuid, chmodsugid,** and **chown** authorizations are explained in "Starting daemons on a trusted system" in the "Customizing system startup" appendix of this guide, and in "Assigning kernel authorizations" in the "Maintaining system security" chapter of this guide.

Kernel authorizations and administrative users

If you are operating with the High and Improved security defaults, you must assign certain kernel authorizations along with subsystem authorizations. Although most of these are already assigned by default, they are listed in Table 4.5 in case you modify the defaults. One exception is the audit subsystem, which requires the addition of the **configaudit** and **suspendaudit** authorizations. These authorizations should never be assigned by default, or to ordinary users.

NOTE Under the Low and Traditional security defaults, most kernel authorizations are already assigned by default.

Table 4-5 Subsystem kernel authorization requirements

Subsystem Authorization	Kernel Authorization Required
audit	configaudit, execsuid, writeaudit
auth	execsuid, chown
backup	execsuid
lp	chown
cron	chmodsugid, chown, execsuid
sysadmin	chmodsugid, chown, execsuid

Adding login shells and configuration files

It is possible to add files for additional login shells that can be selected for users when they are created using **sysadmsh**. For example, **csh**(C) has prototype *.login* and *.cshrc* files that are installed in a user's directory when **csh** is selected as that user's login shell. Each shell has a directory of these prototype files in */usr/lib/mkuser*. You can examine the existing files and follow their example. Make sure the permissions and ownership are consistent with the other files. When you install the files, the new shell is a valid selection in the "Login shell" field of the user Identity form.

Chapter 5

Managing filesystems

This chapter describes one of the most important responsibilities of a system administrator: creating and maintaining filesystems. General maintenance activities are described, such as strategies for maintaining free space. The concept of "filesystem integrity" is introduced, with a description of how the operating system repairs damaged filesystems. Filesystem creation is discussed in the chapter entitled "Adding hard disks and CD-ROM drives." For information on file permissions and other security considerations, see the chapter entitled "Maintaining system security."

What is a filesystem?

A filesystem is a distinct division of the operating system, consisting of files, directories, and the information needed to locate and access them. A filesystem can be thought of as a structure upon which directories and files are constructed.

Each UNIX system has at least one filesystem on the primary hard disk. This filesystem is called "the root filesystem" and is represented by the symbol " / ". The root filesystem contains the programs and directories that comprise the operating system. On small hard disks, the root filesystem also includes all the user directories. The primary hard disk can also be divided into more than one filesystem. This is described in the *Installation Guide.* One of the most common divisions is the */u* filesystem, used to isolate user accounts from the root filesystem.

A UNIX system can also have other filesystems that contain special directories and application programs. Dividing the primary hard disk into multiple filesystems protects the data and makes maintenance easier. Adding still more filesystems by installing other hard disks expands the system storage space. New filesystems can be specifically created by the system administrator, then "attached" (mounted) and "detached" (unmounted) when needed, in the same way that a floppy disk is accessed. This process is described in the chapter entitled "Adding hard disks and CD-ROM drives."

Mounting and unmounting a filesystem

The **mount**(ADM) command attaches a filesystem. For example, to mount or unmount */dev/u* on */u*, you would use the following two commands, respectively:

mount /dev/u /u

Δ **sysadmsh** users select: Filesystems ➪ Mount

umount /dev/u

Δ **sysadmsh** users select: Filesystems ➪ Unmount

Only the super user can use the **mount** and **umount** commands.

NOTE Files in a filesystem are not accessible unless the filesystem is mounted. If files are copied to or created in the mount point directory while the filesystem is unmounted, those files will appear to be in that filesystem when they are not. When the filesystem is mounted, these files seem to "disappear" when the filesystem is mounted over them.

Permitting users to mount filesystems

Only the super user can use the **mount** command. However, the super user can set up parameters to define which filesystems can be mounted by users with the **mnt**(C) command. These parameters may include an access password, if desired.

Each filesystem must have an entry in the file */etc/default/filesys*. Example 5-1 contains a sample set of entries.

Example 5-1 Sample /etc/default/filesys file

```
bdev=/dev/root  cdev=/dev/rroot  mountdir=/  \
desc="The Root Filesystem"  rcmount=no  mount=no

bdev=/dev/u  cdev=/dev/ru  mountdir=/u  rcmount=yes  \
fsckflags=-y  desc="The User Filesystem"

bdev=/dev/x  cdev=/dev/rx  mountdir=/x  mount=yes  \
rcmount=yes  fsckflags=-y  desc="The Extra Filesystem"
```

The sample entries determine the behavior shown in Table 5.1.

Table 5-1 Filesystem mount specifications

Filesystem	When Mounted	Can User Mount?
root	boot time	no
/u	multiuser	no
/x	anytime	yes

If you wish to make any non-root filesystem mountable by users, simply add "mount=yes" to the entry for the given filesystem. In addition, when the **mnt** command is invoked without an argument (no filesystem name), the program checks all non-root filesystems to see if they can be mounted and, if so, mounts them. The option "mount=prompt" asks the user if they want to mount each filesystem where a mount is permitted.

For more information on the **mnt** command, including a complete list of options, refer to the **mnt**(C) manual page.

Filesystem types

Your system can be set up with five different types of filesystem:

- XENIX
- UNIX system
- DOS
- AFS (Acer Fast Filesystem)
- EAFS (Extended Acer Fast Filesystem)

The internal structure of XENIX and UNIX system filesystems differ slightly, but this is of no serious consequence. The AFS (Acer Fast Filesystem) is a faster variant of the XENIX and UNIX system filesystems. The default filesystem is the EAFS (Extended Acer Fast Filesystem), a newer version of the AFS type that supports long filenames (filenames exceeding 14 characters) and symbolic links (file links across filesystems). The block sizes for each of these filesystems are 1K. DOS filesystems are discussed in the chapter entitled "Using MS-DOS and other DOS operating systems."

Converting AFS filesystems to EAFS

An AFS filesystem can be converted to an EAFS filesystem. This is done using the **-E** option of the **fsck**(ADM) command, which normally checks and repairs filesystems. The **-E** option also changes the format of the filesystem to EAFS.

NOTE You must unmount a filesystem before running **fsck**.

The command has the form:

```
fsck -E device
```

where ***device*** is the name of the filesystem device in */dev*.

Converting UNIX filesystems to AFS

A UNIX system filesystem can be converted at any time to an AFS filesystem. This is done using the **-C*clustersize*** option of the **fsck**(ADM) command, which normally checks and repairs filesystems. The **-C** option alters the clustersize to change the format of the filesystem to AFS. The ***clustersize*** argument must be a power of 2 and less than 16 (8 is the recommended value).

The real benefits of the AFS filesystem are seen with a new filesystem. The increase in speed that is possible with a converted AFS filesystem is not immediately apparent; it takes effect only as new files are added to the filesystem. There is little or no benefit in transforming a filesystem that is nearly full; if it is within a few blocks of being full, the conversion does not work. (See the section on "Filesystem integrity" for a complete discussion of **fsck**.)

NOTE You must unmount a filesystem before running **fsck**.

The command has the form:

```
fsck -s -Cclustersize device
```

where ***device*** is the name of the filesystem device in */dev*. Note that the -s option must also be present.

Configurable filesystem features

There are certain filesystem features that are configurable, including:

- Group ID of newly created files: you can specify whether a new file has the group affiliation of the parent directory or that of the creating user.
- Filename truncation: you can specify what happens when an attempt is made to create a filename longer than the system limit.

This section explains these features.

Setting directory SGID bit

By default, the GID (group identifier) of a newly created file is set to the GID of the creating process or user. This behavior can be changed by setting the SGID bit on a directory. Setting the SGID bit on a directory results in a new file having the GID of that directory. To set the SGID bit on a directory, enter the following command, where ***directory*** is the directory name:

```
chmod g+s directory
```

To remove the bit, replace the " + " with a " - " in the above command.

Setting filename truncation

By default, attempts to create filenames longer than the system limit result in the error message "Filename too long." This can be changed so that long filenames are silently truncated to the system limit. The system limit depends on the filesystem type. XENIX, S51K, and AFS filesystems have a limit of 14 characters. The EAFS (default) filesystem has a limit of 255 characters. The default behavior is mandated by POSIX FIPS requirements and is controlled by the **ETRUNC** kernel parameter. This parameter can be changed by invoking the **sysadmsh** selection System ➪ Configure ➪ Kernel ➪ Parameters and selecting category 3: "Files, Inodes and Filesystems," and changing the value of **ETRUNC** to 1. The kernel must then be relinked and booted for the new behavior to take effect. Use the **sysadmsh** selection System ➪ Configure ➪ Kernel ➪ Rebuild to relink the kernel. See the "Reallocating kernel resources with configure" section of the "Tuning system performance" chapter in this guide for complete instructions.

Using links

A link is a directory entry referring to a file. The same file can have several links. This allows a given file to appear wherever it is required without the need for separate files. Any changes made to the file are effectively independent of the name by which the file is known. This means that no matter which link is modified, the same file is sourced. Hard links cannot be made across filesystems and cannot refer to a directory (see the "Symbolic links" section).

The syntax for a hard link is as follows:

ln ***file target***

where ***file*** is the name of the existing file, and ***target*** is the name of the new directory entry which will source the same data. The long file listing generated by the **l**(C) command looks like this example:

```
-rw-rw-r--  2 stevem   pub  60091 Feb  7 19:54  help.file
```

The number "2" that follows the file permissions indicates the number of links. Note that there is no way to distinguish a hard link to a file from its original directory entry.

Symbolic links (EAFS filesystems only)

Symbolic links allow you to connect files or directories on different filesystems. The **-s** option to the **ln**(C) command allows you to do this.

The syntax is as follows:

ln -s ***file target***

You can easily recognize a symbolic link using the long file listing generated by the **l**(C) command:

```
lrwxrwxrwx   1 stevem   group   13 Feb 10 15:34 text -> /u/forbin/file1
```

The arrow "->" points literally to the location of the actual file. Unlike hard links, the source file is always discernable from its links.

If ***target*** is a directory, then one or more files are linked to that directory. If ***source*** is a directory, a directory link is made.

Maintaining free space in filesystems

An important task of the system administrator is filesystem maintenance, which includes keeping the system running smoothly, keeping the filesystems clean, and ensuring adequate space for all users. To maintain the filesystems, the system administrator must monitor the free space in each filesystem, and take corrective action whenever the free space gets too low.

This section explains the filesystem maintenance commands. These commands report how much space is used, locate seldom-used files, and remove or repair damaged files.

A UNIX system operates best when at least 15% of the space in each filesystem is free. In any system, the amount of free space depends on the size of the disk containing the filesystem and the number of files on the disk. Because every disk has a fixed amount of space, it is important to control the number of files stored on the disk.

If a filesystem has less than 15% free space, system operation usually becomes sluggish. If no free space is available, the system stops any attempts to write to the filesystem. This means that the user's normal work on the computer (creating new files and expanding existing ones) stops.

The only remedy for a filesystem that has less than 15% free space is to delete one or more files from the filesystem. The following sections describe strategies for keeping free space available.

Strategies for maintaining free space

The system administrator should regularly check the amount of free space on all mounted filesystems and remind users to keep their directories free of unused files. You can remind users by including a reminder in the */etc/motd* (message of the day) file.

In addition, the **cleantmp**(ADM) command is run by the system to clean the */tmp* directory. You can edit the file */etc/default/cleantmp* to define which, and how often, key directories (*/tmp* and */usr/tmp* by default) are cleaned of files. See the **cleantmp**(ADM) man page for details.

If the amount of free space slips below 15%, the system administrator should follow these steps:

1. Send a system-wide message asking users to remove unused files.
2. Locate exceptionally large directories and files, and send mail to the owners asking them to remove unnecessary files.
3. Locate and remove temporary files and files named *core.*
4. Clear the contents of system log files.
5. Reduce disk fragmentation by making a complete backup of the filesystem, removing all the files, and then restoring them from the backup.
6. If the system is chronically short of free space, it may be necessary to create and mount an additional filesystem.
7. **compress**(C) large infrequently used files.

Displaying free space

You can find out how much free space exists in a particular filesystem with the **df** (for "disk free") command. This command displays the number of "blocks" available on the specific filesystem. A block is 512 characters (or bytes) of data.

The **df** command has the form:

df *specialfile*

Δ **sysadmsh** users select: System ➪ Report ➪ Disk

where *specialfile* can be the name of a UNIX system special file corresponding to the disk drive containing the filesystem. If you do not give a special filename, then the free space of all normally mounted filesystems is given.

For example, to display the free space of the root filesystem */dev/root*, enter:

```
df /dev/root
```

Press ⟨Return⟩. The command displays the special filename and the number of free blocks.

You can find the percentage of free space to total space on your system with the command:

```
df -v
```

Sending a system-wide message

If free space is low, you can send a message to all users on the system with the **wall** (for "write to all") command. This command copies the messages you enter at your terminal to the terminals of all users currently logged in.

To send a message, enter:

```
wall
```

Press ⟨Return⟩. Enter the message, pressing ⟨Return⟩ to start a new line if necessary. After you have entered the message, press ⟨Ctrl⟩**d**. The command displays the message on all terminals in the system.

Displaying disk usage

You can display the number of blocks used within a directory by using the **du** command. This command finds excessively large directories and files.

The **du**(C) command has the form:

du ***directory***

The optional ***directory*** must be the name of a directory in a mounted filesystem. If you do not give a directory name, the command displays the number of blocks in the current directory.

For example, to display the number of blocks used in the directory */u/johnd*, enter:

```
du /u/johnd
```

Press ⟨Return⟩. The command displays the name of each directory in the */u/johnd* directory and the number of blocks used.

Use the **-a** or **-f** options to display files.

Displaying blocks by owner

You can display a list of users and the number of blocks they own by using the **quot** (for "quota") command. The command has the form:

```
quot specialfile
```

The ***specialfile*** must be the name of the special file corresponding to the filesystem.

For example, to display the owners of files in the filesystem on */dev/u*, enter:

```
quot /dev/u
```

Press ⟨Return⟩. The command displays the users who have files in the filesystem and the numbers of blocks in these files.

Mailing a message to a user

If a particular user has excessively large directories or files, you can send a personal message to the user with the **mail** command.

To begin sending a message through the mail, enter:

```
mail login-name
```

where ***login-name*** is the login name of the recipient. The **mail** command copies the message to the user's mailbox, where it can be viewed by the user via the **mail** command. See the *User's Guide* for details.

Locating files

You can locate all files with a specified name, permissions setting, size, type, owner, or last access or modification date using the **find** command. This command is useful for locating seldom-used or excessively large files, or files owned by a particular user.

The **find** command has the form:

```
find pathname option
```

The ***pathname*** is the pathname of the directory that you want to search. The **find** command searches recursively, downward through all the directories under the named directory, for files that match the criteria specified by ***option***. Some options also indicate certain actions for **find** to take on located files. See the **find**(C) manual page for complete details.

Table 5.2 describes some of the options to **find** that are useful for system administrators.

Table 5-2 Useful find options

Option	Description
-atime *+number*	locates files that have not been accessed for the specified *number* of days.
-exec *cmd*	locates files that match the specified criteria and executes *cmd* on those files. The command argument ({}) is replaced by the current pathnames of the files that **find** located. An escaped semicolon (\;) must follow the *cmd* {} construction.
-group *gname*	searches for files that belong to the group *gname*. If *gname* is numeric and does not appear in */etc/group*, it is interpreted as a group ID.
-name *file*	searches for files with the specified name.
-ok *cmd*	works like **-exec** except that it displays the generated command line in prompt form; *cmd* is executed only if the user enters **y** at the prompt.
-perm *onum*	locates all files with permissions that exactly match *onum* (the octal number used with **chmod**(C)).
-print	displays the locations of any files that **find** locates.
-size *+number*	searches for files larger than the specified number of blocks (512 bytes per block).
-type *x*	locates files of a specific type; for example, type **d** for directory or **f** for a file. See **find**(C) for an explanation of the different types.
-user *uname*	locates all files that belong to the user *uname*. If *uname* is numeric and does not appear in */etc/passwd*, it is interpreted as a login ID.

NOTE If you do not include the **-print** option, **find** does not display the list of files that match the search criteria.

Finding temporary files

To locate and display all files named *temp* recursively in the */usr* directory, use the following command:

```
find /usr -name temp -print | more
```

Finding files of a certain size

You can use **find** to locate files of a specified size or type. For example, to locate and print a list of all the directories (**-type d**) greater than 3 blocks in size (**-size +3**) in all the directories (/ and below), enter:

```
find / -type d -size +3 -print
```

Finding files by permissions

Using the **-perm** ***onum*** option to **find**, you can locate all files with permissions that exactly match ***onum*** (the octal number used with **chmod**(C)). For example, to locate and display all the files in the */u* directory that give all users read, write, and execute permissions (***onum*** **is 0777**), enter the following command:

```
find /u -perm 0777 -print
```

Executing commands based on find output

Using the **-exec** option, you can execute a specific shell command on the files that **find** locates. The most common use of **-exec** is to locate a group of files and then remove them. For instance, when you retire a user, you can use **find** to locate all the files owned by that user, back them up, and then remove them from the system. To do this with **find**, enter:

```
find / -user edwarda -print | cpio -ovBc > /dev/rfd096

find / -user edwarda -exec rm "{}" \;
```

The first command locates all the files owned by user *edwarda* and copies the files to an archive. The second command locates the files and then removes them. For more information on copying files to an archive, see the **cpio**(C) manual page.

You can instruct **find** to display a prompt (the command line that **find** generates) before executing the shell command on each file, by using the **-ok** option in place of **-exec**. When you use **-ok**, **find** prompts you with the generated command line:

```
<rm ... /u/edwarda/billboard >?
```

To execute the command (in this case, **rm**), enter **y**.

The command is not executed if you enter a character other than **y**, or if you press ⟨Return⟩.

Another common use of **find** with the **-exec** option is to locate all the files that belong to a particular group and change them. For example, if a user changes groups, you can use **find** to locate and change all their files to the new group:

```
find / -user edwarda -exec chgrp moms "{}" \;
```

If you retire a user and you want to transfer ownership of their files to another user, use the following command:

```
find / -user edwarda -exec chown earnestc "{}" \;
```

Using this construction to execute a command on a large group of files can be very slow because the **-exec** option forks a separate process for each file in the list. A more efficient method for doing this is to use **xargs**(C) in place of **-exec**. The **xargs** command forks fewer processes to execute the command on the entire group of files.

The following example illustrates how to use the **xargs** construction with **find**:

```
find / -user edwarda -print | xargs chown earnestc
```

This command accomplishes the same thing as the previous example, only much more efficiently.

NOTE If the syntax for the command that you want to execute with **xargs** deviates from the standard order (***command options arguments***), you must use **-exec**.

Locating core files

One useful application of **find** is to use the **-name** option to locate core and temporary files for removal.

A core file contains a copy of a terminated program. A UNIX system sometimes creates such a file when a program causes an error from which it cannot recover. A temporary file contains data created as an intermediate step during execution of a program. This file may be left behind if a program contained an error or was prematurely stopped by the user. The name of a temporary file depends on the program that created it. In most cases, the user has no use for either core or temporary files, and they can be safely removed.

When searching for core or temporary files, it is a good idea to search for files that have not been accessed for a reasonable period of time. For example, to find all core files in the */usr* directory that have not been accessed for one week (**-atime +7**), enter:

```
find /usr -name core -atime +7 -print
```

Once you locate the core files, you can remove them easily using one of the following two commands:

```
find /usr -name core -atime +7 -exec rm "{}" \;

find /usr -name core -atime +7 -print | xargs rm
```

These commands find all the core files in the */usr* directory that have not been accessed in the last seven days and remove them.

Checking and clearing log files

A UNIX system maintains a number of log files that contain information about system usage. When new information is generated, the system appends it to the appropriate log file, preserving the file's previous contents. Because some log files can rapidly become quite large, it is important to check the files periodically and, if necessary, clear them by deleting their contents. It may be necessary to retain the most recently appended information in some log files, such as */etc/ddate*. In these cases, trim the files by deleting the previous data and leaving the last entry.

Some log files only grow in special situations. For example, if you want to record all attempts to use the **su** command, and you set the **SULOG** option in */etc/default/su*, the log file */usr/adm/sulog* may build up rapidly.

Table 5.3 lists some of the log files that are most likely to need clearing or trimming. Your system may have different log files from those listed, depending on its configuration and the utilities and application software installed. You may also need to check the files more or less frequently than indicated in the table, depending on system activity. Use the **find**(C) command with the **-size** flag to locate your system's unlisted large log files.

Table 5-3 Administrative log files

Logfile Path	Purpose	Checking Frequency
/etc/wtmp	Historical login record	weekly
/usr/adm/pacct	Process accounting log file	weekly
/usr/adm/messages	System messages log file	weekly
/tcb/audittmp/*	Audit system temporary files	weekly
/etc/ddate	Records date of each backup	monthly or yearly
/usr/spool/uucp/LOGFILE	Records of UUCP job requests, file transfers, system status	monthly
/usr/spool/uucp/.Log/.Old/*	Old UUCP log files stored by **uudemon.clean**	monthly
/usr/spool/lp/logs/request	Record of print requests	automatic†

† Maintenance of this file (and other files) can be accomplished automatically; see "Automatic cleaning: the print request log."

To clear a log file and retain file permissions:

1. Copy the file to a new filename. For example:

   cp /usr/adm/messages /usr/adm/messages.old

2. You can clear the file by using one of the following commands:

   Bourne or Korn shell:

   > /usr/adm/messages

   C shell:

   cat /dev/null > /usr/adm/messages

To monitor new information that is currently being appended to system log files, use the **tail**(C) command as follows:

tail -f /usr/adm/messages

With the **-f** (follow) option, **tail** prints the last ten lines of the file, followed by any lines that are appended to the file between the time **tail** is initiated and killed with the ⟨Del⟩ key.

Automatic cleaning: the print request log

It is also possible to automate the clearing of log files. Following is an example using the */usr/spool/lp/logs/requests* file. This and other files can be cleaned out periodically using, for instance, the **cron** and **crontab** facilities.

This is the default **crontab** entry suggested with the print service:

```
13 3 * * * cd /usr/spool/lp/logs; if [ -f
requests ]; then /bin/mv requests xyzzy; /bin/cp
xyzzy requests; >xyzzy; /usr/lbin/agefile -c2
requests; /bin/mv xyzzy requests; fi
```

It is one line in **crontab** but is split into several lines here for readability. What this entry does, briefly, is "age" the file, changing the name to *requests-1* and moving the previous day's copy to *requests-2*. The number 2 in the **-c** option tells the **agefile** program (this is not a program supplied with the UNIX system) to keep the log files from the previous two days, discarding older log files. By changing this number, you can change the amount of information saved. On the other hand, if you want the information saved more often or want to clean out the file more often than once a day, change the time when the **crontab** entry is run by changing the first two numbers. The current values, 13 and 3, cause the cleanup to occur at 3:13 AM each day.

The default **crontab** entry supplied is sufficient to keep the old print request records from accumulating in the spooling filesystem. For additional information on the request log, see "Using the information in the request log" in the "Using printers" chapter of this guide.

Maintaining efficient filesystem organization

There are two aspects of filesystem usage that degrade the efficiency of filesystems:

- Disk fragmentation (scattering of available disk space) due to constant use and reuse of filesystem blocks.
- Directories growing too large, thus increasing search time for files.

This section explains how to avoid these problems and maintain efficient filesystem organization.

Disk fragmentation

If your system has been in use for some time, the constant creation and removal of files creates a situation called *disk fragmentation*. This means that the files in the filesystem are written in small pieces on the hard disk. This increases access time and reduces system efficiency.

You can reduce disk fragmentation by first making a complete backup of all the files in the filesystem and then removing all the files from the hard disk and restoring them from the backup. For further information, see "Reducing disk fragmentation" in the chapter entitled "Troubleshooting your system."

Because the files are completely rewritten on the disk during the restore, each file is written in one piece and fragmentation is reduced. A small amount of space is also recovered. It is a good idea to perform this action about once a year on a heavily used system and less often on a lightly used system. Be certain that you have complete, accurate, and readable backups before you begin or your files may be lost.

Large directories

It is wise not to allow directories to grow larger than necessary. You should be aware of several size limitations. A directory that contains entries for up to 62* files (plus the required `.` and `..` entries) fits in a single disk block and can be searched very efficiently. A directory can have up to 638* entries and still be viable, as long as it is used only for data storage; anything larger is usually wholly unsuitable as a working directory. It is especially important to keep login directories small, preferably one block at most.

* These figures apply to filenames of 14 characters or less. As filename lengths increase, up to a maximum of 255 characters, the number of files that fit on a single disk block decreases, thus reducing the optimum number of files in a directory.

If large numbers of files are moved in and out of a directory, as in the case of a spool directory, it can also become oversized. This is because, as a rule, directories never shrink. Even if you delete files until you have less than 62 in your login directory, or less than 638 in a spool directory or other data collection directory, system searches of the directory slow down. This is because the directory slot remains the same size as it was prior to the removal of the files.

To see if a directory is too large, even if it does not seem to have over 62 files in it, use the command **hd . | wc -l**. This tells you the number of lines (inodes) in the inode listing.

The solution to the large directory problem is twofold. Educate users to keep their login directories small, and use **cpio** to back up and restore any directory that has an excess of inode numbers left over from deleted or moved files.

Limiting login directory size

To limit the size of the login directory, each user should create a hierarchy of subdirectories that reflects the use or purpose of the contained files. This can make finding files easier for the user. Emphasize the importance of setting up a simple, clear hierarchy, with only five to 10 directories at each level, so files can be added in the future without having to change the hierarchy or move files around.

Removing excess inode numbers

To remove "shadow" inode numbers (numbers left when files are deleted or moved to subdirectories), use **cpio** to back up and restore an oversized directory. Then use **hd** to list the files in the directory and see if there are any that do not show in a standard listing. These "shadow" files appear at the end of the list, and have null inode numbers; 00 and 00 appear in the first two columns of hex digits. Use the following procedure to remove these empty file slots from the directory. Note that the same procedure can be applied to a spool directory.

1. Move to the directory above the oversized one and create a backup directory. For example, if a user named "tracy" has an oversized home directory, enter:

   mkdir tracy.old

2. Move to the oversized directory and back it up:

   find . -print | cpio -pdlm ../tracy.old

   This command copies the directory and all subdirectories, linking files instead of copying them where possible. Confirm that the ownership, group, and permissions of the new directory and files match the original.

3. Use **l** and **hd** to check the backup directory to see that all the files are there and all the shadow inode numbers are gone.

4. Move to the directory above, then delete the oversized directory:

   ```
   rm -rf tracy
   ```

5. Rename the backup directory to replace the oversized one:

   ```
   mv tracy.old tracy
   ```

The directory appears the same to the user, but it can be searched more quickly. This can improve system performance, especially if the procedure is used on numerous working directories.

Adding disk space and filesystems

If free space is chronically low, it may be to your advantage to expand the system's storage capacity by adding a second hard disk as described in the chapter entitled "Adding hard disks and CD-ROM drives." Once the new disk is mounted, you can use the free space in the new filesystem for your work, or even copy user or system directories to it.

Suppose free space is low on your primary hard disk because one filesystem on it is full. If there are others with free space, or there is unused space on the hard disk, you can change the layout on the primary disk, but this is not as simple as adding a second hard disk. It is always best to plan the layout of your hard disk in advance as described in the *Installation Guide.*

A chronic shortage of space usually results from having more users on the system than the current hard disk can reasonably handle, or having too many directories or files. In either case, creating a new filesystem on a new hard disk allows some of the users and directories to be transferred from the primary hard disk, freeing a significant amount of space on the existing filesystem and improving system operation.

If you decide to change the number of filesystems on your primary hard disk or to reapportion the disk space among the filesystems, you must back up your system and reinstall it as described in the *Installation Guide.* You should use the "Fully Configurable" disk initialization so that you can manually control the layout of your disk. During the installation process, use manual layout control to reapportion your disk space as desired.

Filesystem integrity

The most important job of the operating system is to maintain the integrity of filesystem data. Actual loss of data is a rare occurrence; UNIX system filesystems are very resistant to corruption. This is because a certain amount of redundancy exists in special structures that are invisible to the user. It is these structures that ensure filesystem integrity. For example, when the system suffers a power outage, very little information is lost. Any damage usually affects one or two files, making them inaccessible. In almost all cases, the operating system can repair any damage done to files. In very rare cases, damage causes the entire filesystem to become inaccessible.

The operating system uses the **fsck** (for "filesystem check") program to repair damaged filesystems. The **fsck** program checks the consistency of filesystems. In cases where the contents of a file are lost (rare), the only way to restore lost data is from filesystem backups. **fsck** is automatically run at boot time after an abnormal shutdown on the root filesystem. The **fsck** status messages look like this:

```
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Pathnames
** Phase 3 - Connectivity
** Phase 4 - Reference Counts
** Phase 5 - Check Free List
```

If the system terminated abnormally (for example, power outage), you see other messages that may seem alarming:

```
FREE INODE COUNT WRONG IN SUPERBLK
FIX?
```

In fact, this kind of message is routine when a system was not shut down properly, and you only have to enter **y** and **fsck** continues its recovery work. This could be done without the system administrator's intervention, but it is generally better to know what is happening to a filesystem after a problem has occurred.

To discuss the idea of filesystem integrity and how **fsck** functions, it is useful to explain the basic structure of files, directories, and filesystems. Although it is not necessary to understand the principles of file storage, it is helpful to know what the messages like the one above refer to so they will seem less mysterious.

The following section describes some of the basic principles of UNIX operating systems. The section "Repairing a filesystem with fsck" explains the simple mechanics of using the **fsck** command.

How UNIX systems maintain files

Each filesystem contains special structures that allow the operating system to access and maintain the files and data stored on the filesystem. It is the disruption and repair of these structures that we are concerned with.

The structure of a filesystem is based on the way that hard disks store data. Although the hard disk contains all the data used by the system, it is not stored in neat little locations that correspond to individual files. It is unlikely that you could point to a spot on a hard disk and truthfully say: "My file is stored right there on this part of the disk." In fact, the data is probably scattered across the disk. The operating system uses a sophisticated addressing scheme to access each of the pieces that a file is broken into and to present the data to the user as a unit.

The data is spread around because the operating system does not really deal with files, but rather with units of data. For example, assume you created a file and it is actually stored on one part of the disk. Then, suppose you edit that file and delete a few sentences here and there. This means that you are now using a little bit less disk space than when you started. This space amounts to a series of gaps in the area where your file was stored. Disk space is a precious commodity and is not wasted. So, those small amounts of disk space are allocated to other files. Picture this process on a scale of hundreds of files with a dozen users and you have an idea of how files are maintained. Because of the effectiveness of the algorithms (formulas) that the operating system uses, this process is remarkably efficient and trustworthy.

How UNIX systems maintain filesystems

A filesystem contains files and directories that are represented by special structures called "inodes" and "data blocks." These structures make it possible for the operating system to create and keep track of filesystems.

Data blocks	A block is a 1024-byte unit of data stored on the disk. A data block can contain either directory entries or file data. A directory entry consists of an inode number and a filename.
Inodes	An inode can be thought of as a card from a library card catalog. Each inode contains information about a file, just as a card contains information about a book, including its location, its size, the type of file, and the number of directory entries linked to the file. One important point to remember is that an inode does not contain the name of a file; directories contain the actual names. An inode contains the locations of all the data that make up a file so the operating system can collect it all when needed.

Blocks are not just stored on the hard disk. To minimize seeking data on the hard disk, recently used data blocks are held in a cache of special memory structures called buffers. These structures make the operating system more efficient. When enough data is accumulated to write out one or more full disk blocks, the buffer is "flushed" by writing its information to the disk. A minor amount of information is always lost when an outage occurs because recently changed data has not been written to the disk.

With a hard disk filled with data, inodes, directories, files, and blocks cached in memory, how does the operating system keep track of them? The answer is that all these structures maintain sufficient connectivity between files and directories to allow severed connections to be reconstructed.

One special data block, the "super" block, contains overall information about the filesystem, rather than where a particular piece of a file is located. The super block contains the information necessary to mount a filesystem and access its data. It contains the size of the filesystem, the number of free inodes, and information about free space available.

Information is read from the disk version of the super block when the filesystem is mounted and is maintained and modified in memory as activity takes place on the system. The information is written back to the disk at regular intervals by the **init** command which is always running. The **init** command calls the **sync**(ADM) command every 60 seconds, which forces the memory version of the super block and buffers to be written to disk. If the system crashes and the information stored on the disk is not reasonably up-to-date, the filesystem might be corrupted.

Causes of filesystem corruption

Corruption can affect all the structures mentioned in this section. This means that the data or the structures used to locate data can be damaged. This can occur for several reasons:

Hardware Failure	Hardware failures are rare. The best way of dealing with it is to be sure that recommended diagnostic and maintenance procedures are followed conscientiously.
Program Interrupts	It is possible that errors that cause a program to fail might result in the loss of some data. It is not easy to generalize about this because the range of possibilities is so large.
Human Error	While it may be painful to admit it, probably the greatest cause of filesystem corruption falls under this heading. There are rules that should be followed when managing filesystems.

Rules for checking filesystems

When checking and repairing filesystems, keep in mind the following rules:

- To make sure users cannot access the filesystem you are planning to unmount, bring the system down to single-user (maintenance) mode before unmounting it.
- Unmount the filesystem with **umount**(ADM) before checking and repairing it.
- Unmount the filesystem, such as one on a floppy disk, before physically removing it.
- If you plan to make the filesystem available, remember to remount the filesystem after checking and repairing it.

Regular filesystem backups represent the best assurance of continued filesystem integrity.

Repairing filesystems with fsck

To check and repair filesystems, use **fsck**(ADM). The **fsck** command examines the various structures on the disk and attempts to reconcile them. Where possible, **fsck** reestablishes connections and resolves references; it "cleans" the filesystem.

Before repairing a filesystem with **fsck**, shut down the system and bring it up in single-user (maintenance) mode. To do this without completely shutting down the system, use the **su** argument to **shutdown**:

```
/etc/shutdown -g10 su
```

The first argument indicates the number of minutes before system shutdown.

NOTE The **shutdown** command calls **sync** automatically.

For more information on shutting the system down, see **shutdown**(ADM).

Once the system is in single-user mode, unmount the filesystem with **umount**(ADM). Check and repair the filesystem using the following command:

fsck ***filesystem***

Δ **sysadmsh** users select: Filesystems ⇨ Check

where ***filesystem*** is the name of the special file corresponding to the device name of the filesystem. For example, the */dev/u* device file corresponds to the */u* filesystem.

To check the root filesystem, use the **-b** argument to **fsck** :

fsck -b /dev/root

The system must be in single-user mode before running "**fsck -b**" on the root filesystem. This command automatically remounts the root filesystem after checking it.

NOTE The **fsck** program is actually a front-end that invokes a version of **fsck** for each filesystem type. For example, **fsck** calls a special version to repair DOS filesystems.

For example, if you bring your system back up in single-user mode after a power failure, use the **fsck** command to check the */u* filesystem before going into multiuser mode. To do this, enter the following command:

fsck /dev/u

The **fsck** program checks the filesystem and reports on its progress with the following messages:

```
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Pathnames
** Phase 3 - Connectivity
** Phase 4 - Reference Counts
** Phase 5 - Check Free List
```

If a damaged file is found during any one of these phases, **fsck** asks if it should be repaired or salvaged. Enter **y** to repair a damaged file. You should always allow the system to repair damaged files even if you have copies of the files elsewhere or intend to delete the damaged files.

Note that the **fsck** command deletes any file that it considers too damaged to be repaired. You can elect for **fsck** to make the repair or not. You might choose to have **fsck** ignore an inconsistency because the problem is so severe that either you want to fix it yourself using the **fsdb**(ADM) utility, or you plan to restore your system from backups. If you cannot use **fsdb**, you must allow **fsck** to resolve the inconsistencies or the filesystem may not be usable. (**fsdb** is described in "Repairing a filesystem when fsck stops at size check" in the chapter entitled "Troubleshooting your system.")

You may need to run **fsck** several times before the entire filesystem is clean. For a complete list of error messages, see the **fsck**(ADM) manual page.

Summary of fsck phases

The **fsck** program scans and examines each of the structures mentioned earlier. Each phase compares components and checks that these components agree with each other.

Phase 1 checks the blocks and sizes. The **fsck** program reads the inode list to determine the sizes and locates the blocks used by each file. Inodes are checked for inode type, zero link counts, inode sizes, and bad or duplicate blocks. (Bad blocks are block values outside the boundaries of a filesystem.) When **fsck** asks whether or not to clear an inode, this means to zero out the bad information in the inode. This removes the file or directory that was associated with it. A duplicate block means that two inodes point to the same block on the disk. The **fsck** command attempts to find the original inode along with the duplicate for correction in phase 2.

Phase 2 checks the pathnames. Files removed in phase 1 must then have their directory entries removed. Phase 2 cleans up error conditions caused by improper inode status, out-of-range inode pointers, and directories that point to bad inodes as described earlier. For files with duplicate blocks found in phase 1, **fsck** wants to remove both files (this is one of the few areas where system administrator intervention is useful).

Phase 3 checks for connectivity. Phase 2 removed directories that do not point to valid files. Phase 3 reconnects files that were severed from the directory structure. Any files that are unreferenced but valid are placed in a special directory called *lost+found.* Because the directory was severed, the name of the file is lost and a number is assigned to the file in *lost+found.*

NOTE **fsck** does not create or extend the *lost+found* directory. There must already be a sufficient number of empty slots in the directory for use by **fsck** when reconnecting files. A *lost+found* directory is created automatically by **mkdev fs** when a new filesystem is created.

Phase 4 checks the reference counts. The **fsck** command checks the link count of each entry that survives phases 2 and 3. In some cases, files that were not pointed to under the directory structure, but still have an inode, can be relinked to the filesystem in *lost+found.*

Phase 5 checks the free list. The **fsck** command examines the free-block list maintained by the filesystem and resolves the missing or unallocated blocks allocated or removed earlier. When an inconsistency is detected, **fsck** prompts to rebuild it.

Phase 6 salvages the free list. If specified in phase 5, the system reconstructs a free block list from the altered filesystem.

Automatic filesystem check

The operating system sometimes requests a check of the filesystem when you first start it. This usually occurs after an improper shutdown (for example, after a power loss). The filesystem check repairs any files disrupted during the shutdown.

Chapter 6 *Adding multiport cards, memory, and other bus cards*

The bus (or "motherboard") of your computer is the center of your system. Every system administrator must deal with the bus and the hardware associated with it. To find the bus on your system, you must generally remove the shell from the main body of your computer. Generally, you find a large circuit board with expansion slots for extra boards. These boards are commonly known as "bus cards."

Bus cards can be extra memory for your system, host adapters, multiport serial boards for extra terminals, controller boards for peripheral devices such as hard disks, tape drives, control cards for monitors with color and graphics capabilities, mouse controllers, or other devices. In this chapter we explain a little about bus cards and how to install them in your UNIX system. Installation of most devices with bus cards is explained in detail in other chapters of this guide.

Installing bus cards

To install a bus card, you must first shut down the operating system and power down the system. Make sure that the computer is unplugged or you can injure yourself. Before you begin working on the computer, ground yourself by touching a metal object close at hand that is not the computer. Static electricity that builds up and jumps from your hand when you touch the hardware inside the computer can ruin your equipment.

NOTE Micro Channel (MCA) and Extended Industry Standard Architecture (EISA) bus cards do not have dip switches and jumpers; use the Reference Diskette (for MCA machines) or EISA Configuration Diskette to adjust settings.

In addition, EISA hardware can be configured to be either "edge triggered" (AT-style non-shareable interrupts) or level-triggered (MCA-style shareable interrupts). Depending on the driver available, you may want to configure the card to act one way or the other.

Dip switches and jumpers

Before you plug your board into the bus, make sure that there are no settings on the board that must be changed. Again, your hardware documentation that comes with the board should list the default settings and how to change them. Generally, to change the settings of a board, there are dip switches and "jumpers". Dip switches operate in "down" and "up" positions. Your hardware documentation should list the correct settings if your board has these switches. Jumpers are clips that slide over metal posts that stick out of the board to make a connection. You can change the settings on a board by moving the jumper to connect a different pair of posts. Again, your hardware documentation should provide you with specific instructions for jumper settings on your hardware.

NOTE Your UNIX system is designed to work with most hardware using default settings. You will rarely have to change the settings on a board.

Installing the hardware

Carefully perform any steps necessary to expose the expansion slots on your computer. Your hardware documentation should explain this in detail. Once you can examine this area, note the number of available spaces for bus cards. A new system has up to 8 or 10 available slots. Note that some slots are longer than others. There are both short and long cards. Short cards are about half as long as long cards. Usually there are two to three short slots and the rest are long slots. Find a slot that fits your board and gently, but firmly, plug the board into the slot in the bus. The board should have a tab on one side that fits into the slot on the bus. Bus cards only fit one way.

Some bus cards have a port that should face the outside of the computer. There may be a small plate covering an opening in the computer held on with a small screw. You can remove this cover plate if you need to. Boards such as modems, serial and parallel cards, and external device control cards require this.

When you are done, replace the shell for your computer, and turn it on and boot. You may first need to use the manufacturer's setup program as described below to change the system's configuration before you can use the new hardware.

NOTE The **eisa**(ADM) utility can be used to list cards currently installed on the EISA bus. The **slot**(C) utility does the same thing for MCA machines.

Using the manufacturer's setup diskette

Many machines, particularly Micro Channel (MCA) and Extended Industry Standard Architecture (EISA) machines, include a manufacturer's setup program on a bootable floppy disk. (Some machines have a setup program in ROM or "hidden" on the hard disk.) Copy this disk for use and keep the original in a safe place. This disk configures the permanent memory on your computer to describe the system hardware setup. Whenever you add a major device, like an extra hard disk or an extra serial card, you may need to run your setup program to tell your computer about the new hardware. Some computers automatically recognize the presence of new hardware. Your manufacturer's documentation should let you know if you need to run this software.

Adding more memory

You can improve system performance and run larger programs by increasing the amount of internal memory.

To increase internal memory follow these steps:

1. Turn off your computer. Steps for this task are provided in the "Starting and stopping the system" chapter of this guide.
2. Install extended memory according to the manufacturer's instructions. Make sure you have set all switches as noted in the instructions.
3. Boot the operating system. The boot screen details how the additional memory has affected your system.

Many system resources depend on the amount of memory installed. For example, the "kernel i/o bufs" displayed at boot time are determined by the **NBUF** kernel parameter. When this parameter is set to zero, the number of kernel buffers is determined at boot time based on the amount of memory installed. For more information on system resources related to RAM, see "Adding memory (RAM)" in the "Tuning system performance" chapter of this guide.

If the memory hardware reports an error, the following message is displayed:

```
PANIC: memory parity error
```

You then see the software reboot message:

```
      ** Safe to Power Off **
               - or -
  ** Press Any Key to Reboot **
```

If the system repeatedly panics from parity errors, consider replacing the memory chips.

NOTE Some machines have a hardware limitation on the maximum amount of memory that can be installed. Refer to your computer hardware manual to determine the maximum amount of memory you can install.

The UNIX system uses only "extended", not "expanded", memory.

Adding and configuring standard serial ports

To add a multiport expansion card, you must first determine whether the card is a "smart" serial card or a standard serial card. If the card is a "smart" card, the manufacturer will have supplied installation software and a driver. These should be all you need to add the card to your UNIX system. Before installing your card, check your *Release Notes* for information about hardware compatibility. Follow the instructions for insertion furnished with your card, referring to your computer hardware manual if necessary. If you are using a supported 4- or 8-port expansion board, check to see if your board is recognized at bootup by checking the UNIX system bootup message. If the boot process does not accurately report your board, then the switches on your card are not set properly. Check your board's hardware documentation for the proper switch settings and the *Release Notes* for the correct addresses. This applies to boards that are listed as supported in the *Release Notes.*

Vendor-supplied drivers may not print a recognition message at boot time. If your serial expansion card is a smart card with a vendor-supplied driver, you should not need to run **mkdev serial** to install it. For your system to recognize the new card, run the vendor-supplied installation software.

Configure the interrupts for the two standard COM ports: COM1 as interrupt 4 and COM2 as interrupt 3. Most serial cards use one interrupt per board, so two four-port boards can use COM1 and COM2. Be aware of the requirements of other products and hardware to avoid interrupt conflicts. See **serial**(HW) for more information on COM1 and COM2.

NOTE You cannot use the COM3 and COM4 serial ports because there are only two interrupt vectors in the IBM interrupt scheme allocated to COM devices.

Make certain you first configure your hardware according to the manufacturer's instructions. If your system includes a configuration diskette or BIOS setup program, use it as instructed. If your system is configured with switch settings on the main system board (motherboard), define the new ports by setting the proper switches (refer to your hardware manuals for the settings).

If your card is a standard serial card, the following instructions explain how to create new device files for additional ports:

1. Boot the system and enter system maintenance mode.
2. When you are in system maintenance mode, enter:

   /etc/mkdev serial

   Δ **sysadmsh** users select: System ⇨ Hardware ⇨ Card_Serial
3. The following is displayed:

```
Serial Board Initialization

You would like to install a:
        1. 1 port card
        2. 2 port card
        3. 4 port card
        4. 5 port card
        5. 8 port card
        6. 16 port card

Enter your choice or q to quit:
```

   Enter the appropriate number and press ⟨Return⟩.
4. The program responds with the following menu (only COM1 and COM2 appear and are usable on most systems):

```
The card is configured as:
        1. COM1
        2. COM2
        3. COM3
        4. COM4

Enter your choice or q to quit:
```

5. Enter a number and press ⟨Return⟩. After **mkdev** accepts the COM slot, you see:

```
Which card do you have (the following are supported)?

1. card      base address     0xnnn
2. card      base address     0xnnn
3. card      base address     0xnnn
4. card      base address     0xnnn
5. card      base address     0xnnn
6. card      base address     0xnnn
7. card      base address     0xnnn

Enter your choice or q to quit:
```

   Select the number that corresponds to the card you have installed and press ⟨Return⟩.

6. You are asked if you want the default baud rate on the modem lines:

```
Would you like the modem devices set at the default speed of 1200. (y/n)
```

   If you respond **y**, the default is used. If you respond **n**, you are asked to provide a baud rate.

7. When the process is complete, you see messages similar to the following:

```
Modifying system files...
System files have been successfully updated.

The following standard serial device(s) have been installed:
     ttyxa ttyxb ttyxc ttyxd ...

The following device(s) access the same physical port as
their lower case counterparts, but have modem control properties:
     ttyxA ttyxB ttyxC ttyxD ...
```

8. You see the following:

```
You must create a new kernel to effect the driver change you specified.
Do you wish to create a new kernel now? (y/n)
```

   Unless you wish to make additional changes, you should respond **y** and press ⟨Return⟩.

9. Next, you see:

```
     The UNIX operating system will now be rebuilt.
     This will take a few minutes.  Please wait.

     Root for this system build is /.
```

   As part of the linking process, you see the following messages:

```
     The UNIX kernel has been rebuilt.
Do you want this kernel to boot by default? (y/n)
```

   Answer **y** if you want this kernel to be used every time you boot the system.

10. The following is displayed:

```
Backing up /unix to /unix.old.
Installing new /unix.
The kernel environment includes device node files and /etc/inittab.
The new kernel may require changes to /etc/inittab or device nodes.
Do you want the kernel environment rebuilt? (y/n)
```

   Enter **y** and press ⟨Return⟩.

11. The following is displayed:

```
The new kernel has been successfully linked and installed.
To activate it, reboot your system.
Setting up new kernel environment.
```

   The kernel is now configured with the additional serial port(s).

12. Use the **shutdown**(ADM) command (or **haltsys**(ADM) if you are in single-user mode) to shut down the system and reboot.

NOTE An error message is displayed if you attempt to access a serial port that is not installed and defined.

Adding and configuring parallel ports

The system configures one parallel port automatically (parallel port #1, */dev/lp0*). If you install more than one parallel port, you must use the **mkdev parallel** command to configure it properly. Table 6.1 lists the default addresses and interrupts associated with the parallel ports.

Table 6-1 Parallel port defaults

Name	Device	Address	Interrupt
Serial/parallel adapter #1	/dev/lp0	0x378	7
Monochrome adapter	/dev/lp1	0x3bc	7
Serial/parallel adapter #2	/dev/lp2	0x278	5 (7 on MCA)

NOTE ISA machines cannot share interrupts.

To configure an additional parallel port, do the following:

1. Boot the system and enter system maintenance mode.
2. When you are in system maintenance mode, enter:

   mkdev parallel

   Δ **sysadmsh** users select: System ➪ Hardware ➪ Parallel

   You see the following menu:

```
Do you wish to:

        1. Add a parallel port
        2. Remove a parallel port

Select an option or enter q to quit:
```

   Enter **1** and press ⟨Return⟩.

3. You are then asked to choose the port:

```
Which port would you like to add ?

        1. Serial/parallel adapter #1
        2. Monochrome adapter
        3. Serial/parallel adapter #2

Select an option or enter q to quit:
```

   Any port already configured is not shown on the menu. For example, if adapter #1 is already configured, only items 2 and 3 appear. Enter the number of the port you want to install and press ⟨Return⟩.

4. Each port you selected is then added to the kernel (the default settings are used). You see:

```
You must create a new kernel to effect the driver change you specified.
Do you wish to create a new kernel now? (y/n)
```

   Unless you wish to make additional changes, you should respond **y** and press ⟨Return⟩.

5. Next, you see:

```
     The UNIX operating system will now be rebuilt.
     This will take a few minutes.  Please wait.

     Root for this system build is /.
```

   As part of the linking process, you see the following messages:

```
     The UNIX kernel has been rebuilt.

Do you want this kernel to boot by default? (y/n)
```

   Answer **y** if you want this kernel to be used every time you boot the system.

6. The following is displayed:

```
Backing up /unix to /unix.old.
Installing new /unix.

The kernel environment includes device node files and /etc/inittab.
The new kernel may require changes to /etc/inittab or device nodes.

Do you want the kernel environment rebuilt? (y/n)
```

   Enter **y**.

7. The following is displayed:

```
The new kernel has been successfully linked and installed.
To activate it, reboot your system.

Setting up new kernel environment.
```

   The kernel is now configured with the additional parallel port(s).

8. Use the **shutdown**(ADM) command (or **haltsys**(ADM) if you are in single-user mode) to shut down the system and reboot.

You can invoke **mkdev parallel** at any time to remove or add new ports.

Chapter 7
Using printers

Printers are a useful addition to any computer system. Most systems require the ability to print out data on paper. A wide variety of printing hardware or line printers are supported. Some line printers are *parallel* devices, but most are connected as *serial* devices.

To add a printer, the system administrator must:

- connect the physical hardware to the computer, then
- use the correct system commands to enable the printer for operation.

This chapter explains how to do this and how to maintain printers once they are added. Note that physical connections between a printer and the system vary depending on hardware configuration. This chapter provides some information about making the necessary physical connections, but for more information about these connections, see the hardware manuals provided with the printer and your computer.

The operating system supports serial printers which use the standard RS-232 interface and parallel printers which use the Centronics parallel interface. To find out which interface your printer uses, check your hardware documentation.

The printer spooling system

The UNIX system line printer spooling system is a collection of commands that help you, as system administrator, to install, monitor, and efficiently control the line printers serving your system. A request to print a file is *spooled* or lined up with other printing jobs to be sent to the printer. Each print job is processed and waits its turn in line to be printed. This line of pending print jobs is called the *queue.*

When a user requests a file to be printed using the **lp**(C) command, the line printer system responds with a "request ID." This consists of the name of the printer on which the file is printed and a unique number identifying the file. With this request ID, the user can find out the status of the print request or cancel it. The **lp** options help the user to control printer output easily.

The print service performs the following functions:

- handles the task of receiving files that users want printed
- filters the files (if necessary) so they can print properly
- schedules the work of one or more printers
- starts programs that interface with the printer(s)
- keeps track of the status of jobs
- alerts you to printer problems
- keeps track of the mounting of forms
- issues error messages when problems arise

There are several terms used in this chapter to describe the operation of the print service:

device	The target for **lp** output. A device is represented by a full pathname of a special device file.
printer	The name assigned by the system administrator to represent a device. This name can have up to 14 characters. At different times, a printer can be associated with different devices.
class	An ordered list of printers. Print requests sent to a class of printers are printed by the first available member of that class.
destination	A place where print requests are sent. A destination can be a class or a printer.

Consult your computer and line printer hardware manuals for information on making the connection between your system and printing devices.

Installing a printer

This section instructs you on how to install new printing devices on your UNIX system. You must connect the printer to a proper port (serial port for serial printers, parallel port for parallel printers), ensure that it works, and set up the UNIX system printer spooling software using the **sysadmsh** Printers selection.

NOTE If you have any problems connecting your printer, see the chapter in this guide entitled "Troubleshooting your system" for helpful information.

Follow the steps below to install a printer:

1. Find a place for your printer and make sure that it is properly assembled and plugged into a power outlet.

2. **If you are connecting a serial printer:** connect the RS-232 cable from a computer serial port to the port on your printer. Serial printers must be capable of supporting XON/XOFF or DTR (Data Terminal Ready) protocols and must be configured for those protocols. Consult your printer owner's manual for more information. Next, enter the following command substituting the correct port number for ***nn***:

   disable /dev/tty*nn*

   Press ⟨Return⟩. This disables logins on the port you have connected to your printer and allows the port to be used for serial communication.

3. **If you are connecting a parallel printer:** the printer must use a standard Centronics interface cable. The main parallel port (built-in as opposed to being on a monochrome card) should be configured for interrupt vector 7 and is automatically recognized as **lp0**. You must run **mkdev parallel** if you plan to use a port other than the built-in parallel port. For more information on running **mkdev parallel**, refer to the chapter entitled "Adding multiport cards, memory, and other bus cards" in this guide. For information on configuring your parallel ports, consult your hardware manual.

4. Verify that you have connected the printer correctly by sending data directly to the device. (This procedure is shown in flowchart form in Figure 7-1 later in this section.) Enter one of the following commands:

   For serial printers:

   date > /dev/tty*nn*

   where ***nn*** is the number of the serial port you are using (for example, */dev/tty1a*).

   NOTE When sending output directly to a serial printer you may need to specify some **stty** settings. For example:

   (stty 4800 ; date >/dev/tty1a) </dev/tty1a

   This would be used for testing output on a 4800 baud printer.

For parallel printers:

date > /dev/lp*n*

where ***n*** is the number of the parallel port you are using (for example */dev/lp0*).

NOTE For certain laser printers (for example: HP LaserJet) you need to send a form feed to the printer. Use the following command, where ***xxx*** is the tty (serial) or lp (parallel) port:

(date ; echo "\f\c") > /dev/*xxx*

5. If you do not see the date printed on your printer, the most likely cause is some type of hardware malfunction. The following troubleshooting procedures may help you to isolate the problem.

   For serial printers:

   - Make certain you are using the non-modem control device, for example: */dev/tty1a,* not */dev/tty1A.* (For more information on the naming convention for serial ports, see **serial**(HW).)
   - Try using a cable with only pins 2, 3, and 7 connected.
   - Try using a cable with only pins 2, 3, and 7 connected, with pins 2 and 3 swapped.
   - Recheck your printer configuration by verifying its switch positions in your printer hardware manual.
   - Make certain that the system recognizes your serial port. You can verify this by running the **hwconfig**(C) command, or by checking the file */usr/adm/messages.* If your port is one of the non-intelligent boards supported by the built-in serial driver, you see a message similar to the following:

     ```
     %serial 0x03F8-0x03FF 04 - unit=0 type=Standard nports=1
     ```

     If "unit=0" is displayed, the serial port is considered to be COM1. If the unit is 1, the port is considered to be COM2. "nports=" denotes how many ports the driver recognized on the board. If you connect your serial printer to the first port on COM1, the associated device name will be tty1a. The second device on COM1 is tty1b, and so on. Devices on COM2 are named tty2a, tty2b, etc.

Intelligent serial boards using third-party drivers may display different bootup messages specific to their drivers; they may also use a different scheme for device names. The message displayed in this case may look similar to the following:

```
%ONBOARD  0x0230-0x023F 34  0 unit=0 mem=0x000D0000 nport=16
```

Be sure to read the documentation for the board and its drivers before attempting to install serial devices such as printers.

- Recheck the switch settings on your serial port. If you are using a multiport card, try other lines on that card and be sure it does not conflict with the standard COM ports.
- Try attaching the printer to a standard serial port, COM1 or COM2, to see if the printer and cabling are functioning correctly.

For parallel printers:

- Make certain your cable is securely connected and all wires are sound. A good way of testing this is to use the cable on another system that is known to be working correctly.
- Recheck your printer configuration by verifying its switch settings in your printer hardware manual.
- Recheck the switch settings on your parallel card; it must also be recognized at bootup. You can verify this by running the **hwconfig**(C) command, or by checking the file */usr/adm/messages* for a message similar to the following:

```
parallel 0x378-0x37A   07    -    unit=0
```

To confirm that your card is recognized, enter the following command:

hwconfig name=parallel

If the card is recognized, an entry will be printed that has similar information to the entry above. The interrupt vector is listed in column 3 (or with "vec=" in the case of the **hwconfig** display); make certain that it does not conflict with other hardware.

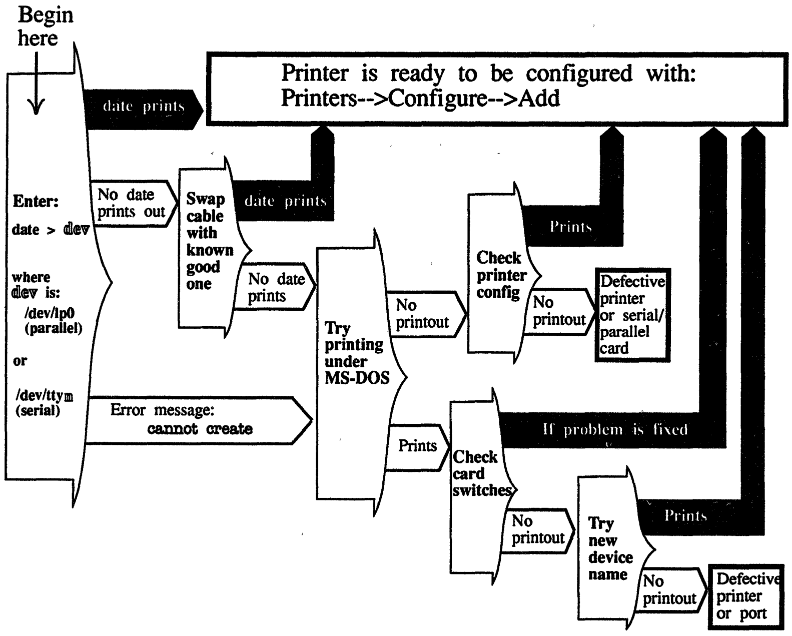
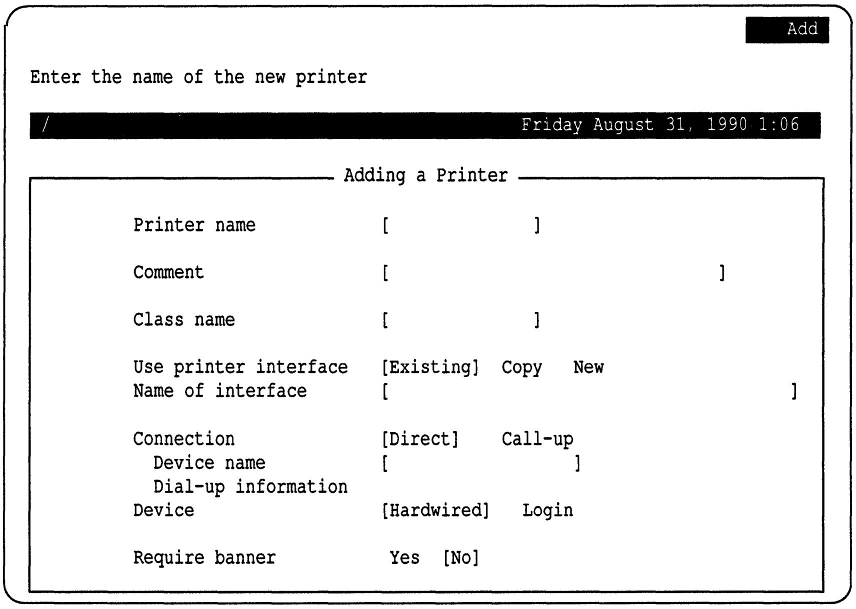


Figure 7-1 Testing your printer connection

6. You should now run **sysadmsh**. The **sysadmsh** Printers selection displays a form with a series of fields that you must fill in. If you make a mistake while responding to the questions, just press the ⟨Esc⟩ key and start again. To begin, make the following **sysadmsh** selection:

   Printers ➪ Configure ➪ Add

The following form is displayed (the fields are discussed below):



```
                                                                    Add

Enter the name of the new printer

/                                              Friday August 31, 1990 1:06

                        Adding a Printer

          Printer name           [              ]

          Comment                [                                ]

          Class name             [              ]

          Use printer interface  [Existing]  Copy   New
          Name of interface      [                                       ]

          Connection             [Direct]    Call-up
            Device name          [                ]
            Dial-up information
          Device                 [Hardwired]   Login

          Require banner          Yes  [No]
```

Here is an explanation of each field:

Printer name	Name of the new printer (limit: 14 characters)
Comment	Comment that describes the printer
Class name	Name of the class associated with this printer (⟨F3⟩ for list)
Use printer interface	Use an existing interface program, a copy or new printer interface. (Available printer interfaces are described under "Interface program" in the next section, "Printer configuration options")
Name of interface	Name of the interface program (or ⟨F3⟩ to list existing interfaces)
Connection	Whether the printer is directly connected to the computer, or must be called up via a modem or network
Device name	Name of device to which the printer is connected (for example, */dev/tty1a* for a serial printer and */dev/lp0* for a parallel printer)
Dial-up information	Modem phone number or network system name
Device	Whether connection is dedicated to the printer, or is also used for a login terminal (will be disabled by scheduler)

Require banner	Force a banner (a page with the printer name, date, and so forth) to always be printed or allow the user to request no banner be printed.

When you finish filling in the form, (press ⟨Return⟩ on the Require banner field), it is executed and the new configuration set is put in place. To use the printer, you must also start the print service, enable the printer and allow the printer to accept requests. Do this with the following **sysadmsh** selections:

Printers ➪ Schedule ➪ Begin
Printers ➪ Schedule ➪ Enable
Printers ➪ Schedule ➪ Accept

In the case of the **Enable** and **Accept** selections, you must supply the name of the printer when prompted to do so.

Additional configuration information can be supplied for your printer; this is discussed in the section entitled "Printer configuration options."

For more information on printer maintenance commands, see the sections entitled "Starting and stopping the print service," "Managing the printing load," and "Enabling and disabling printers." The **sysadmsh** includes all these functions, supplementing the **lpadmin**(ADM) command.

Printer configuration options

Although the default values for printer configuration are usually sufficient for most needs, there are a number of options to configure individual aspects of printer operations. These include such options as fault alerting and recovery. The following is a list of additional information that can be given to define the configuration of each printer:

- interface program
- printer type
- content types
- connection method
- character sets or print wheels
- fault alerting
- fault recovery
- use restrictions
- default printing attributes

You need to give very little of this information to add a new printer to the print service; however, the more information you provide, the better the printer is managed for you and the more efficiently the print service will run.

The descriptions in the following sections help you understand what this printer configuration information means and how it is used so that you can decide how to configure your printers. In each section, you are also shown how to specify this information when adding a printer. While you can follow each of the sections in order and correctly configure a printer in several steps, you may want to wait until you have read all of the sections before adding a printer so that you can do it in one step.

Interface program

Interface programs (also known as interface scripts) are used to initialize printers and take advantage of their particular capabilities. The *standard* interface program defines minimal capabilites common to most printers. These programs are human-readable files located in */usr/spool/lp/model*. The interface programs already present on your system are listed in Table 7.1. You can select an interface program when you install or change a printer configuration using the **sysadmsh** selections

Printers ➪ Configure ➪ Add

or

Printers ➪ Configure ➪ Modify

Table 7-1 Printer interface programs

Name	Description
1640	DASI 1640 terminal
5310	AT&T 5310/20 Matrix Printer
HPDeskJet500	Hewlett Packard DeskJet 500
HPDeskJetplus	Hewlett Packard DeskJet Plus
HPLaserJet	Hewlett Packard LaserJet
TandyDMP	Tandy DMP Printers
crnlmap	serial or parallel printer requiring newline mapped to control-linefeed
dqp10	DQP-10 Matrix Printer
dumb	dumb line printer
emulator	Tandy Printers in IBM Emulation Mode
epson	Epson serial or parallel printer

(Continued on next page)

Table 7-1 Printer interface programs
(Continued)

Name	Description
f450	DASI 450 terminal
hp	hp2631a line printer
lqp40	LQP-40 Letter Quality Printer
network	remote printing over UUCP or ethernet
network.ps	remote printing over UUCP or ethernet for PostScript printers
ph.daps	Autologic APS-5 phototypesetter
postscript	PostScript printer
pprx	Printronix line printer with parallel interface
proprinter	IBM Proprinter XL
prx	Printronix line printer
qume1155	Qume Sprint 1155 line printer
standard	standard printer interface program
ti800	Texas Instruments 855 printer

Many of these interface programs have special options available by using the **-o** option to the **lp** command. (Read the appropriate interface file for this information.) For example, the *PostScript* interface program includes the options listed in Table 7.2.

Table 7-2 PostScript options

Option	Description
port	prints text in portrait mode
land	prints text in landscape mode
land2	prints text in 2 page landscape mode
raw	prints a PostScript file

You can also create your own interface scripts or customize existing ones to suit your needs. See the section entitled "How to write an interface program" later in this chapter.

Printer type

The printer type is the generic name for the printer. The print service uses the printer type to extract information about the printer from the *terminfo* database. This information describes the capabilities of the printer so that you can be warned if some of the configuration information you provide is not appropriate for the printer. The information also describes the control data to use to initialize the printer before printing a file. While you are not required to specify a printer type, you are advised to specify one so that better print services are provided.

The printer type is the generic name for the printer. Typically, the printer type is derived from the manufacturer's name, such as "495" for the AT&T 495 Laser Printer. Examine */usr/lib/terminfo* for an appropriate entry for your "type" of printer. There might be one entry specific to your printer, or one that is generic to a type. Specify the printer type as follows:

/usr/lib/lpadmin -p ***printername*** **-T** ***printer-type***

Δ **sysadmsh** users select: Printers ➪ Configure ➪ Parameters

If you do not define the printer type, the default **unknown** is used. This produces empty results when the print service looks up information about the printer, so the print service cannot verify certain requests or initialize the printer.

Content types

While the printer type information tells the print service what type of printer is being added, the content type information tells the print service what types of file can be printed. Most printers can print only one type of file; for them, the content type is likely to be identical to the printer type. Some printers, however, can accept several different types of file and print their contents properly. When adding this kind of printer, you should list the names of the content types it accepts.

When a file is submitted to the print service for printing, the print service searches for a printer capable of handling the job. The print service can identify an appropriate printer through either the content-type name or the printer-type name. Therefore, you can specify either name (or no name) when submitting a file for printing.

Content-type names may look a lot like printer-type names, but you are free to choose names that mean something to you and the people using the printer. (The names *simple, terminfo,* or *any* are recognized as having particular meanings by the print service; be sure to use them consistently.) The names must contain no more than 14 characters and may include only letters, digits, and underscores. If the same content type is printable by several different types of printer, you should use the same content type names when you add those printers. This makes it easier for the people using the printers because they can use the same name to identify the type of file they want printed regardless of the printing destination.

For example, several manufacturers produce printers that accept PostScript files. While these printers may need different printer types so that each can be properly initialized (assuming the initialization control sequences are different), they may all be capable of handling the same type of input file, which you call, perhaps, *PostScript.* As another example, several manufacturers produce printers that accept ANSI X3.64 defined escape sequences. However, the printers may not support all the ANSI capabilities or may support different sets of capabilities. You may want to give different content-type names for these printers to differentiate them.

You do not have to list the content types for a printer. If you do not, the printer type is used as the name of the content type the printer can handle. If you have not specified a printer type, the print service assumes the printer can print only files of content type *simple.* This may be sufficient if users pick the proper printer and make sure the files are properly prepared for the printer before they are submitted for printing.

The most common type of file on the UNIX system is known as *simple.* This file is assumed to contain just printable ASCII characters and the following control characters:

backspace	Moves the printing mechanism back one space, except at the beginning of a line
tab	Moves the printing mechanism to the next tab stop, which is normally every 8 columns on most printers
linefeed	Moves the printing mechanism to the beginning of the next line (may require special port settings for some printers – see the next section "Printer port characteristics")
form feed	Moves the printing mechanism to the beginning of the next page
carriage return	Moves the printing mechanism to the beginning of the same line (may fail on some printers)

The word "carriage" may be archaic for modern laser printers, but similar actions apply. If a printer can handle a *simple* type of file, you should include it in the content type list when you add the printer and specify the content type(s) that the printer can handle. If you do not want a printer to accept files of type *simple*, you must give an alternate list of content types that the printer can accept. (The printer type is a good name to use if no other type is appropriate.)

Another content type name is *terminfo*. This does not refer to a particular type of file but instead refers to all the types represented in the *terminfo* database. It is not likely that any printer is capable of handling all the types listed in the database. However, this name is reserved for describing possible filter capabilities. Likewise, the content type *any* is reserved for describing the types of files a filter can accept or produce. These names should not be used as content types when adding a printer.

Specify the list of content types as follows:

> **/usr/lib/lpadmin -p*printername* -I *content-type-list***

Δ **sysadmsh** users select: Printers ➪ Configure ➪ Content

The ***content-type-list*** is a list of names separated by a comma or space. If you use spaces to separate the names, enclose the entire list (but not the **-I**) in quotes. If you do not define the types of files a printer can accept, the print service assumes it can take type *simple* and a type with the same name as the printer type (if the printer type is defined).

Connection method

The print service allows you to connect your printers in a variety of ways. The simplest way is to connect your printer directly to the computer. However, you may want to connect printers via a network or through a dialed modem, where they can be shared with other computers or workstations. Once you have connected the printer to the computer or connected it to a network, and connected the network to the computer, you should describe the connection method for the print service.

The default method by which printers are connected to the computer is the direct connection method. If you have used this method to connect your printer to your computer, you generally need to do only one other thing: name the connecting port.

There are two methods of making non-direct connections: through a dial-up modem or over any other type of network. The difference between a UUCP network printer and a dialup printer is that a dialup printer does not go through the UUCP spooling process; the connection is made directly and the print job prints directly on the dialed-up printer. See "Configuring a network printer" or "Configuring a dialup printer" later in this chapter for instructions.

Character sets or print wheels

Printers differ in the way they can print in different font styles. Some have changeable print wheels, some have font cartridges, others have preprogrammed, selectable character sets. The print service, with your help, can minimize the impact of these differences on the users of the print service.

When adding a printer, you can specify what print wheels, font cartridges, or character sets are available with the printer. Only one of these is assumed to apply to each printer. From the point of view of the print service, however, print wheels and font cartridges are the same because they require you to physically intervene and mount a new print wheel or font cartridge. Thus, for ease of discussion, only print wheels and character sets are mentioned.

When you list the print wheels or character sets available, you are assigning names to them. These names are for your convenience and the convenience of the users. Because different printers may have similar print wheels or character sets, you should use common names for all printers. This allows a person to submit a file for printing and to ask for a particular font style, without regard for which printer is used or whether a print wheel or selectable character set is used.

If the printer has mountable print wheels, you need only list their names. If the printer has selectable character sets, you need to list their names and map each one into a name or number that uniquely identifies it in the *terminfo* database. You can use the following command to determine the names of the character sets listed in the *terminfo* database:

TERM=*printer-type* tput csnm 0

printer-type is the name of the printer type in question. The name of the 0th character set (the character set obtained by default after the printer is initialized) should be printed. Repeat the command, using 1, 2, 3, and so on in place of the 0, to see the names of the other character sets. In general, the *terminfo* names should closely match the names used in the user documentation for the printer. However, because not all manufacturers use the same names, the *terminfo* names may differ from one printer type to the next.

NOTE For the print service to find the names in the *terminfo* database, you must specify a printer type. See the earlier section "Printer type."

To specify a list of print wheel names when adding a printer, enter the following command:

/usr/lib/lpadmin -p *printername* -S *print-wheel-list*

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Parameters

print-wheel-list is a list of names separated by a comma or space. If you use spaces to separate the names, enclose the entire list (but not the **-S**) in quotes.

To specify a list of character set names and to map them into *terminfo* names or numbers, enter the following command:

> **/usr/lib/lpadmin -p** ***printername*** **-S** ***character-set-list***

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Parameters

character-set-list is also a list of names separated by a comma or space; however, each item in the list looks like one of the following:

> **cs*N*=*character-setname***
> ***character-setname1=character-setname2***

N in the first case is a number from 0 to 63 that identifies the number of the character set in the *terminfo* database. ***character-setname1*** in the second case identifies the character set by its *terminfo* name. In either case, the name to the right of the equal sign "=" is the name you choose as an alias of the character set.

NOTE You do not have to provide a list of aliases for the character sets if the *terminfo* names are adequate. You can refer to a character set by *terminfo* name, by number, or by your alias.

For example, suppose your printer has two selectable character sets (sets #1 and #2) in addition to the standard character set (set #0). The printer type is 5310. You enter the following commands to determine the names of the selectable character sets:

```
TERM=5310 tput csnm 1
english
TERM=5310 tput csnm 2
finnish
```

The words *english* and *finnish* are the output of the commands, the names of the selectable character sets. You feel that the name "finnish" is adequate for referring to character set #2, but better names are needed for the standard set and set #1. You enter the following command to define synonyms:

> **/usr/lib/lpadmin -p** ***printername*** **-S "cs0=american, english=british"**

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Parameters

If you do not list the print wheels or character sets that can be used with a printer, then the print service assumes the following: a printer that takes print wheels has only a single, fixed print wheel, and people cannot ask for a special print wheel when using the printer. Also, a printer that has selectable character sets can take any **cs*N*** name or *terminfo* name known for the printer.

Alerting to mount a print wheel

If you have printers that take changeable print wheels and you have listed the print wheels allowed on each, then users can submit a print request to use a particular print wheel. However, until it is mounted (see "Mounting a form or print wheel" in this chapter), a request for a print wheel stays queued and is not printed. You could periodically monitor the number of print requests pending for a particular print wheel, but the print service provides an easier way. You can ask to be alerted when the number of requests waiting for a print wheel has exceeded some threshold.

You can choose one of several ways to receive an alert:

- You can receive an alert via electronic mail. See **mail**(C) for a description of the **mail** command.
- You can receive an alert written to whatever terminal you are logged in on. See **write**(C) for a description of the **write** command.
- You can receive an alert through a program of your choice.
- You can receive no alerts.

NOTE If you elect to receive no alerts, you are responsible for checking whether the proper print wheel is mounted.

In addition to the method of alerting, you can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the print wheel is mounted. You can choose the rate of repeated alerts, or you can choose to receive only one alert per print wheel.

To arrange for alerting to the need to mount a print wheel, enter one of the following commands:

/usr/lib/lpadmin -S ***print-wheelname*** **-A mail -Q** ***integer*** **-W** ***minutes***
/usr/lib/lpadmin -S ***print-wheelname*** **-A write -Q** ***integer*** **-W** *minutes*
/usr/lib/lpadmin -S ***print-wheelname*** **-A** ***'command'*** **-Q** ***integer*** **-W** ***minutes***
/usr/lib/lpadmin -S ***print-wheelname*** **-A none**

Δ **sysadmsh** users select: Printers ⇨ Auxiliary ⇨ Alert

The first two commands direct the print service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the print service to run ***command*** for each alert. The shell environment currently in effect when you enter the third command is saved and restored for the execution of ***command;*** this includes the environment variables, user and group IDs, and current directory. The fourth command directs the print service to never send you an alert when the print wheel needs to be mounted. ***integer*** is the number of requests that need to be waiting for the print wheel, and ***minutes*** is the number of minutes between repeated alerts.

NOTE If you want mail sent or a message written to another person, you will have to use the third command listed. Use the **-A 'mail *user-name*'** or **-A 'write *user-name*** option.

Once you start receiving repeated alerts, you can direct the print service to stop sending you alerts for the current case by giving the following command:

/usr/lib/lpadmin -S*print-wheelname* -A quiet

Δ **sysadmsh** users select: Printers ⇨ Auxiliary ⇨ Alert

Once the print wheel is mounted and unmounted again, alerts start again if too many requests are waiting. Alerts also start again if the number of requests waiting falls below the **-Q** threshold and then rises up to the **-Q** threshold again, as when waiting requests are canceled or if the type of alerting is changed.

If ***print-wheelname*** is **all** in any of the commands above, the alerting condition applies to all print wheels for which an alert has already been defined.

If you do not define an alert method for a print wheel, you do not receive an alert for it. If you do define a method but do not give the **-W** option, you are alerted once for each occasion.

Fault alerting

The print service provides a framework for detecting printer faults and alerting you. Faults can range from simple problems, such as running out of paper or ribbon or needing to replace the toner, to more serious faults, such as a local power failure or printer failure. The range of fault indicators is also broad, ranging from dropping carrier (the signal that indicates that the printer is online) to sending an XOFF, or a message. Only two classes of printer fault indicators are recognized by the print service itself: a drop in carrier and an XOFF not followed in reasonable time by an XON. However, you can add filters that can recognize any other printer fault indicators and rely on the print service to alert you to a fault when the filter detects it.

NOTE For a description of how to add a filter, see the "Using forms and filters" section in this chapter. For a description of how a filter should let the print service know a fault has occurred, see the "Customizing the print service" section in this chapter.

You can choose one of several ways to receive an alert to a printer fault:

- You can receive an alert via electronic mail. See **mail**(C) for a description of the **mail** command.
- You can receive an alert written to the terminal on which you are logged in (any terminal). See **write**(C) for a description of the **write** command.

- You can receive an alert through a program of your choice.
- You can receive no alerts.

NOTE If you elect to receive no alerts, you need a way of finding out about the faults and fixing them; the print service does not continue to use a printer that has a fault.

In addition to the method of alerting, you can also arrange for repeated alerts every few minutes until the fault is cleared. You can choose the rate of repeated alerts, or you can choose to receive only one alert per fault.

NOTE Without a filter that provides better fault detection, the print service cannot automatically determine when a fault has been cleared except by trying to print another file. It assumes that a fault is cleared when it successfully prints a file. Until that time, if you have asked for only one alert per fault, you do not receive another alert. If after you have fixed a fault, but before the print service has tried printing another file, the printer faults again, or if your attempt to fix the fault did not succeed, you are not notified. Receiving repeated alerts per fault or requiring manual re-enabling of the printer (see "Fault recovery") overcomes this problem.

To arrange for alerting to a printer fault, enter one of the following commands:

```
/usr/lib/lpadmin -p printername -A mail -W minutes
/usr/lib/lpadmin -p printername -A write -W minutes
/usr/lib/lpadmin -p printername -A 'command' -W minutes
/usr/lib/lpadmin -p printername -A none
```

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Errors

The first two commands direct the print service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the print service to run ***command*** for each alert. The shell environment currently in effect when you enter the third command is saved and restored for the execution of ***command***. The environment includes environment variables, user and group IDs, and current directory. The ***minutes*** is the number of minutes between repeated alerts. The fourth command directs the print service not to send you an alert when a fault occurs.

NOTE If you want mail sent or a message written to another person when a printer fault occurs, use the third command. Use the option:

```
-A 'mail username
```

or

```
-A 'write username
```

Once a fault occurs and you start receiving repeated alerts, you can direct the print service to stop sending you alerts for the current fault by giving the following command:

> **/usr/lib/lpadmin** -p *printername* **-A quiet**

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Errors

If *printername* is **all** in any of the commands above, the alerting condition applies to all printers.

If you do not define an alert method, you receive mail once for each printer fault. If you do define a method but do not give the **-W** option, you are alerted once for each fault.

Fault recovery

Once a printer fault is detected and you are alerted, you will probably fix the fault and get the printer ready for printing. When the printer is ready for printing again, the print service recovers in one of three ways:

- continues printing at the top of the page where printing stopped,
- restarts printing at the beginning of the print request that was active when the fault occurred, or
- waits for you to tell the print service to re-enable the printer.

NOTE The ability to continue printing at the top of the page where printing stopped requires the use of a filter that can wait for a printer fault to be cleared before resuming properly. Such a filter probably has to have detailed knowledge of the control sequences used by the printer so it can keep track of page boundaries and know where in a file printing stopped. The default filter used by the print service cannot do this. If a proper filter is not being used, you are notified in an alert if recovery cannot proceed as you want.

To specify the way the print service should recover after a fault has been cleared, enter one of the following commands:

> **/usr/lib/lpadmin -p** *printername* **-F continue**
> **/usr/lib/lpadmin -p** *printername* **-F beginning**
> **/usr/lib/lpadmin -p** *printername* **-F wait**

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Errors

These direct the print service, respectively, to continue at the top of the page, restart from the beginning, or wait for you to enter an **enable** command to re-enable the printer (see the "Enabling and disabling printers" section later in this chapter for information on the **enable** command).

If you do not specify how the print service is to resume after a printer fault, it tries to continue at the top of the page where printing stopped, or failing that, at the beginning of the print request.

If the recovery is **continue** but the interface program does not stay running so that it can detect when the printer fault was cleared, printing is attempted every few minutes until it succeeds. You can force the print service to retry immediately by issuing an **enable** command.

Restricting user access to a printer

You can limit the use of a printer to a subset of all people on your computer. You may want to do this, for instance, if a printer is being set aside for printing sensitive information and only a subset of the people are allowed to print sensitive information, or if use of a high-quality printer incurs expenses not all people are authorised to incur.

The print service uses the list of users allowed or denied for a printer to restrict use of the printer. The print service refuses a user's request to print a file on a printer he or she is not allowed to use.

The method of listing the users allowed or denied for a printer is similar to the method used to list users allowed or denied access to the **cron** and **at** facilities. Briefly, the rules are as follows:

- An allow list contains those users allowed to use the printer. A deny list contains those users denied access to the printer.
- If the allow list is not empty, the deny list is ignored. If the allow list is empty, the deny list is used. If both lists are empty, there are no restrictions on who can use the printer.
- Putting **any** or **all** into the allow list allows everybody to use the printer; putting **any** or **all** into the deny list denies everybody, except the user **lp** and the super user *root*.

You can add names of users to either list using one of the following commands:

> **/usr/lib/lpadmin -p** ***printername*** **-u allow:*user-list***
> **/usr/lib/lpadmin -p** ***printername*** **-u deny:*user-list***

Δ **sysadmsh** users select: Printers ➪ Configure ➪ Users

user-list is a list of names of users separated by a comma or space. If you use spaces to separate the names, enclose the entire list (including **allow:** or **deny:** but not the **-u**) in quotes. The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list. Using **allow:all** allows everybody; using **deny:all** denies everybody.

If you do not add user names to the allow or deny lists, the print service assumes that everybody can use the printer.

Default printing attributes

When a user submits a request to print a file, the page size, character pitch, and line pitch (that is, print spacing) are normally determined from the form that it is printed on. If the user does not require a form, he or she can give the page size and print spacing to use. However, if the user gives neither a form to use nor the page size and print spacing, defaults are used.

You can set the defaults for each printer. This can also serve to make submitting a print request easier, by designating different printers as having different default page sizes or print spacing. Users then simply route their file to the appropriate printer to get the style output they want. For example, you can have one printer dedicated to printing wide (132 column) output, another printing normal (80 column by 66 lines) output, yet another printing letter quality (12 characters per inch, 8 lines per inch).

You can independently specify four default settings: page width, page length, character pitch, and line pitch. You can scale these to fit your needs. The first two can be given in columns and lines, inches, or centimeters. The last two can be given as characters and lines per inch or per centimeter. In addition, the character pitch can be specified as **pica** for 10 characters per inch (cpi), **elite** for 12 cpi, or **compressed** for the maximum cpi the printer can provide (up to a limit of 30 cpi).

Set the defaults using one or more of the following commands:

```
/usr/lib/lpadmin -p printername -o width=scaled-number
/usr/lib/lpadmin -p printername -o length=scaled-number
/usr/lib/lpadmin -p printername -o cpi=scaled-number
/usr/lib/lpadmin -p printername -o lpi=scaled-number
```

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Parameters

Add the letter "i" to *scaled-number* to indicate inches, or the letter "c" to indicate centimeters. The letter "i" for character pitch (cpi) or line pitch (lpi) is redundant. You can also give **pica, elite,** or **compressed** instead of a number for the character pitch.

If you do not provide defaults, the page size and print spacing are those available when the printer is initialized. You can find out what the defaults are by first defining the printer configuration without providing your own defaults, then using the **lpstat** program to display the printer configuration. The command

```
lpstat -p printername -l
```

reports the default page size and print spacing. If you have not provided the defaults, the reported defaults are calculated from the *terminfo* database entry for the printer. Obviously, this requires you to have provided a printer type in the printer configuration.

Print service command summary

The print service has four user-level commands, which are shown in Table 7.3.

Table 7-3 User print service commands

Command	Description
cancel(C)	Cancels a request for a file to be printed
lp(C)	Sends a file or files to a printer
lpstat(C)	Reports the status of the print service
lprint(C)	Prints from printer attached to a terminal

In addition to sending requests to the print service system, checking the status of requests, and canceling requests, users can be given the ability to disable and enable a printer. The idea is that if a user finds a printer is malfunctioning in some way, it should not be necessary to call the administrator to turn the printer off. On the other hand, it may not be reasonable in your printing environment to allow regular users to disable a printer. You can control whether other users have access to the two commands shown in Table 7.4 by assigning or revoking the *printerstat* authorization (see "Changing user authorizations" in the "Administering user accounts" chapter in this guide).

Table 7-4 Privileged print service commands

Command	Description
disable(C)	Deactivates the named printer(s)
enable(C)	Activates the named printer(s)

A separate set of commands available for the administrator is shown in Table 7.5. These commands are found in the */usr/lib* directory. If you expect to use them frequently, you might find it convenient to include that directory in your **PATH** variable. To use the administrative commands, you must be logged in as either *root* or have the *lp* authorization. See the "Administering user accounts" chapter for an explanation of authorizations. (**lpsched** must be run by *root*.)

Note that all these commands are accessible under the **sysadmsh** Printers selection. You will also probably need to use the commands for disabling and enabling a printer and the rest of the commands described earlier in this section.

Table 7-5 Administrative print service commands

Command	Description
/usr/lib/accept	Permits job requests to be queued for a specified destination
/usr/lib/reject	Prevents jobs from being queued for a specified destination. Described on the same manual page as **accept**(ADM)
/usr/lib/lpadmin	Sets up or changes printer configurations
/usr/lib/lpfilter	Sets up or changes filter definitions
/usr/lib/lpforms	Sets up or changes preprinted forms (Enter /usr/lib/lpadmin to mount a form)
/usr/lib/lpmove	Moves output requests from one destination to another. Described on the same manual page as **lpsched**(ADM)
/usr/lib/lpsched	Starts the print service
/usr/lib/lpshut	Stops the print service. Described on the **lpsched**(ADM) manual page
/usr/lib/lpusers	Sets or changes the default priority and priority limits the users of the print service can request

These commands are also accessed through the **sysadmsh** "Printers" selection, which is much easier than the complex syntax of the print service commands.

Adding a local printer

The operating system also supports the use of local printers attached to the AUX or PRINT port on the back of many serial terminals. These printers are connected via standard RS-232 connections and can significantly reduce the load on shared system resources. The **lprint**(C) command is used to print files on a local printer, but the terminal must be properly configured for the command to work. To add a printer connected to the AUX or PRINT port on the back of a terminal and use it for local printing, follow this procedure:

1. Connect your local serial printer to the AUX port on your terminal with a standard RS-232 cable with pins 2, 3, and 7 (minimum) connected. Make sure the printer is powered on and is online. (If the terminal supports pass-through mode to the parallel port, a parallel port may be used.)
2. Log in to the UNIX system on the terminal and verify that the terminal is working correctly.
3. Make sure that the AUX port on your terminal is configured with the same settings as your printer (baud rate, parity, data bits, xon/xoff, and so forth).
4. For the **lprint** command to work, **lprint** needs to know how to start and stop local printing for each specific terminal. **lprint** looks in the file */etc/termcap* to find two terminal attributes: PN (start printing) and PS (stop printing). These are escape sequences that must be sent to the terminal to control local printing. Very few terminals have these attributes defined in their **termcap** entries. Use a text editor (such as **vi**(C)) to examine the */etc/termcap* file. (*/etc/termcap* can also be an alternate file, as defined by the **TERMCAP** variable.) Search for the entry for your terminal. For example, if your terminal is a Wyse 60, you would search for "wyse60".

   The **termcap** entry for the Wyse 60 appears in Example 7-1.

   Example 7-1 Wyse 60 termcap entry

```
w7|wy60|wyse60|Wyse WY-60 with 80 column/24 line screen in wy60 mode:\
        :is=\E'\072\Ee(\EO\Ee6\Ec41\E~4\Ec21\Ed/:\
        :if=/usr/lib/tabset/std:pt:\
        :G1=\EH3:G2=\EH2:G3=\EH1:G4=\EH5:GD=\EH0:GG#0:GH=\EH\072:\
        :GU=\EH=:GV=\EH6:GR=\EH4:GL=\EH9:GC=\EH8:GF=\EH7:\
        :PU=\EJ:PD=\EK:\
        :al=\EE:am:bs:bt=\EI:cd=\EY:ce=\ET:cl=\E+:\
        :cm=\Ea%i%dR%dC:co#80:dc=\EW:dl=\ER:ei=\Er:im=\Eq:k0=^AI\r:\
        :k1=^A@\r:k2=^AA\r:k3=^AB\r:k4=^AC\r:k5=^AD\r:k6=^AE\r:k7=^AF\r:\
        :k8=^AG\r:k9=^AH\r:kd=^J:kh=^~:kl=^H:kr=^L:ku=^K:\
        :li#24:mi:nd=^L:se=\EG0:so=\EG4:sg#0:ug#0:ue=\EG0:ul:up=^K:us=\EG8:\
        :PN=\Ed#:PS=^T:hs:ts=\Ez(:fs=^M:
```

The Wyse 60 has PN and PS defined. With other terminals, you must add a line containing these two attributes to the */etc/termcap* entry for your terminal. The line you add has the form:

`:PN=`***start sequence*:**`PS=`***stop sequence:***

5. Refer to your terminal manual to find the sequence of control characters used to switch the auxiliary port on and off. This is sometimes referred to as "passthrough" or "transparent" mode. For an example of the sequence to enable auxiliary printing, the code to switch the port on for a Wyse 60 terminal is:

   〈Esc〉 d #

   And the code to turn it off again is:

   〈Ctrl〉**t**

6. These keystrokes must be translated into **termcap** format before inserting them into the **termcap** file. **termcap** uses the codes in Table 7.6 to represent keystrokes.

Table 7-6 termcap keystroke translations

Keystroke	termcap sequence
ESCAPE	\E
CTRL x	^x (x is any character)
NEWLINE	\n
RETURN	\r
TAB	\t
BACKSPACE	\b
FORMFEED	\f

To use a control sequence, use the caret " ^ " symbol, not the 〈Ctrl〉 key. For example, 〈Ctrl〉x would be represented by ^x. In addition, characters can be represented by their octal codes (see **ascii**(M)), and the caret and backslash characters represented by \^ and \\, respectively. Entries for **termcap** attributes must be separated by a colon " : ". (See **termcap**(M) for more details.)

Recall that the **termcap** attributes for starting and ending printing are PN and PS. Using the table above the **termcap** entry for the Wyse 60 keystrokes 〈Esc〉 **d #** (start printing, PN) and 〈Ctrl〉**t** (stop printing, PS) looks like this:

```
:PN=\Ed#:PS=^T:\
```

7. For a terminal missing these entries, you simply insert a modified version of the above line into the **termcap** entry for the terminal. (You must be certain to insert the line within the entry for your terminal; do not add it as the first line or the last line.) For other terminals, check your owner's manual and locate the proper sequences for turning the auxiliary print mode on and off and substitute the **termcap** sequences as in the example. Some terminals (such as the Wyse 60) include a "transparent" mode, where the data is not displayed on the screen as it is printed. (This is the mode selected by the PN sequence in the example.)

   NOTE You must be logged in as *root* to edit */etc/termcap*. We recommend that you copy the original file to another name in case you make an error. You can also extract the file again from your distribution using **custom**(ADM).

8. Once you have added the PN and PS entries, log out and back in again to activate the new *termcap* entry.
9. Use the following command to print the file ***filename*** on your local printer:

   lprint *filename*

   Do not touch your keyboard while local printing is taking place; you cannot perform other tasks on your terminal while printing.
10. If your file is printed on the screen instead of the printer, the PS and PN entries you created are incorrect. Revise the entries with the correct codes. If the file still does not print on the printer or the terminal, try crossing the Transmit and Receive Data pins in the cable connecting the terminal AUX port and the printer. (This is also known as a "null modem" connection.)

    NOTE You can eliminate carriage return delays by setting the environment variable **CRDELAY** to N. This should be done only if you are running a fast printer.

11. If, when printing with **lprint**, everything prints on one line, you should set the environment variable **FORMS** to X. In the Bourne shell (sh), use:

    FORMS=X; export FORMS

    In the C shell (csh), use:

    setenv FORMS X

    These commands may be placed in the */etc/profile* and */etc/cshrc* files respectively, in which case they will affect all users. They also may be placed in individual users' *.profile* and/or *.cshrc* files.

This environment setting turns on **lprint**'s "transparent" mode. In this mode **lprint** does not perform special processing of carriage returns, line feeds, form feeds, or tabs. This means that your terminal's **stty** settings must match those of the printer.

If the terminal and printer require different **stty** settings, a shell script is appropriate (this step is unnecessary in many cases). Enter the following lines as */usr/bin/lprints*:

```
:
# /usr/bin/lprints
#
# Do local printing with stty settings that differ from
# those of the terminal. The required settings are read
# from environment variable LPRINTSTTY.
oldstty='stty -g'
[ ''$LPRINTSTTY'' != '''' ] && stty $LPRINTSTTY
FORMS=X /usr/bin/lprint ''$@''
stty $oldstty
```

chmod 755 */usr/bin/lprints* and execute this file in place of **lprint**.

Each user may now store the **stty** settings required for their local printer in the environment variable **LPRINTSTTY**, or system-wide values may be set in */etc/profile* and */etc/cshrc*, for example:

```
LPRINTSTTY=''ixon ixoff -ixany onlcr''; export LPRINTSTTY
```

Starting and stopping the print service

Under normal operation, you should never have to start or stop the print service manually. It automatically starts each time the system goes into multiuser mode. However, if you need to stop the print service without stopping the operating system as well, you can do so by following the procedure described in the next section.

Stopping the print service causes all printing to cease within seconds. Any print requests that have not finished printing are printed in their entirety after the print service restarts. The printer configurations, forms, and filters in effect when the print service is stopped are restored after it is restarted.

NOTE To start and stop the print service manually, you must be logged in as *root*.

Manually stopping the print service

To stop the print service manually, enter the following command:

/usr/lib/lpshut

Δ **sysadmsh** users select: Printers ⇨ Schedule ⇨ Stop

This message is displayed:

```
Print services stopped
```

All printing ceases within a few seconds. If you try to stop the print service when it is not running, you see the message:

```
Print services already stopped
```

NOTE Jobs can appear to pass through a printer that is not online. If a printer is not online or operating properly, you should disable the printer.

Manually starting the print service

To restart the print service manually, enter the following command:

/usr/lib/lpsched

Δ **sysadmsh** users select: Printers ⇨ Schedule ⇨ Begin

This message is displayed:

```
Print services started
```

It may take a minute or two for the printer configurations, forms, and filters to be re-established before any saved print requests start printing. If you try to restart the print service when it is already running, you see the message:

```
Print services already active
```

NOTE You do not have to stop the print service to change printer configurations or to add forms or filters.

Canceling a print request

To cancel a printout you have requested, use the **cancel**(C) command. When you request a printout, the system displays a request ID for your job. For example, if you send a job to a printer named "laser," the UNIX system displays the request ID as:

```
request id is laser-number
```

where ***number*** is the number assigned to your job. To cancel the job before it begins printing, use the following command:

cancel laser-*number*

Δ **sysadmsh** users select: Printers ⇨ Request ⇨ Cancel

The printout is canceled.

Most systems print quickly, so a **cancel** command must be used promptly to have any effect.

Enabling and disabling printers

The **enable** command allows **lpsched** to print files on printers. A printer can accept requests for printing after the **accept** command is given for it, but to print the files, the **enable** command must be given as well.

For example, to enable a printer named "daisy," enter:

enable daisy

Δ **sysadmsh** users select: Printers ⇨ Schedule ⇨ Enable

You can disable printers with the **disable** command. The scheduler, **lpsched,** does not send printing requests to disabled printers regardless of their status with respect to the **accept** command. The **disable -r** option allows you to send a message to users explaining why a printer was disabled.

For example, to disable a printer named "laser" because of a paper jam, enter:

disable -r "paper jam" laser

Users requesting the status of "laser" with the command **lpstat -plaser** receive the following message:

```
printer laser disabled since Dec 5 10:15
paper jam
```

For more information on these two commands, see the **enable**(C) and **disable**(C) manual pages.

Adding a printer to a class

It is occasionally convenient to treat a collection of printers as a single class. The benefit is that a person can submit a file for printing by a member of a class, and the print service picks the first printer in the class that it finds free. This allows faster turnaround, as printers are kept as busy as possible.

Classes are not needed if the only purpose is to allow a user to submit a print request by type of printer. The **lp -T** *type* command lets a user submit a file and specify its type. The first available printer that can handle the type of file prints the file.

The print service avoids using a filter, if possible, by choosing a printer that can print the file directly over one that needs it filtered first.

One use of classes is to put into a class a series of printers that should be used in a particular order. If you have a high-speed printer and a low-speed printer, for instance, you probably want the high-speed printer to handle as many print requests as possible, with the low-speed printer reserved for use when the other is busy. Because the print service always checks for an available printer in the order that the printers were added to a class, you could add the high-speed printer to the class before the low-speed printer and let the print service route print requests in the order you wanted.

Add a printer to a class using the following command:

/usr/lib/lpadmin -p ***printername*** **-c** ***classname***

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Modify

If the class *classname* does not exist yet, it is created. (Use **lpstat -c** to get a list of all classes and their printers.)

NOTE Class names and printer names must be unique. This allows a user to specify the destination for a print request without having to know whether it is a class of printers or a single printer. Thus, you can not have a class and printer with the same name.

A printer does not belong to any class until you add it to one.

Setting the system default destination

You can define the printer or class used to print a file when the user has not explicitly asked for a particular destination and has not set the **LPDEST** shell variable. The printer or class must already exist first.

Make a printer or class the default destination by entering the following command:

/usr/lib/lpadmin -d ***printername*** or ***classname***

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Default

If you later decide that there should be no default destination, enter a null ***printername*** or ***classname*** as in the following command:

```
/usr/lib/lpadmin -d
```

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Default

If you do not set a default destination, there is none. Users must explicitly name a printer or class in each print request, or they have to set the **LPDEST** shell variable with the name of a destination.

For C-shell:

```
setenv LPDEST printer
```

For Bourne or Korn shells:

```
LPDEST=printer;export LPDEST
```

Users can also place these commands in their *.login* and *.profile* files, respectively.

Mounting a form or print wheel

NOTE See the "Using forms and filters" section in this chapter for information about preprinted forms.

Before the print service starts to print files that need a preprinted form or print wheel, you have to mount it on a printer. (Print wheels are used on older, so-called "daisy wheel" impact printers that use small wheels with the print characters around the perimeter.) If alerting has been set on the form or print wheel, you are alerted when enough print requests are queued for it to be mounted.

When you mount a form, you may wish to see if it is lined up properly. If an alignment pattern is registered with the form, you can ask that this be repeatedly printed until you have adjusted the printer so that the alignment pattern looks correct.

Mounting a form or print wheel involves first loading it onto the printer and then telling the print service that it is mounted. Because it is difficult to do this on a printer that is currently printing and because the print service continues to print files not needing the form on the printer, you will probably have to disable the printer first. Thus, the proper procedure is as follows:

1. Disable the printer using the **disable** command.
2. Mount the new form or print wheel as described later in this section.
3. Re-enable the printer using the **enable** command. (The **disable** and **enable** commands were described earlier in the "Enabling and disabling printers" section of this chapter.)

After loading the new form or print wheel into the printer, enter the following command to tell the print service to mount it. (This command is shown on two lines for readability; it must be entered as one line.)

> **/usr/lib/lpadmin -p** *printername* **-M -S** *print-wheelname*
> **-f** *formname* **-a -o filebreak**

Δ **sysadmsh** users select: Printers ➪ Auxiliary ➪ PPforms ➪ Configure

Leave out **-S** *print-wheelname* if you are mounting just a form, or leave out the **-f** *formname* **-a -o filebreak** if you are mounting just a print wheel.

If you are mounting a form, you are asked to press the ⟨Return⟩ key before each copy of the alignment pattern is printed. After the pattern is printed, you can adjust the printer and press the return key again. If no alignment pattern is registered, you are not asked to press the key. You can drop the **-a** and **-o filebreak** options if you do not want to bother with the alignment pattern.

The **-o filebreak** option tells the print service to add a *formfeed* after each copy of the alignment pattern. The actual control sequence used for the *formfeed* depends on the printer involved and is obtained from the *terminfo* database. If the alignment pattern already includes a formfeed, leave out the **-o filebreak** option.

If you want to unmount a form or print wheel, use the following command:

> **/usr/lib/lpadmin -p** *printername* **-M -S none -f none**

Δ **sysadmsh** users select: Printers ➪ Auxiliary ➪ PPforms ➪ Remove

Leave out **-S none** if you just want to unmount a form; leave out **-f none** if you just want to unmount a print wheel.

Until you mount a form on a printer, only print requests that do not require a form are sent to it. Likewise, until you mount a print wheel on a printer, only print requests that do not require a particular print wheel are sent to it.

Removing a printer or class

You can remove a printer or class if it has no pending print requests. If there are pending requests, you have to first move them to another printer or class using the **lpmove** command, or remove them using the **cancel** command.

Removing the last remaining printer of a class automatically removes the class as well. However, the removal of a class does not cause the removal of printers that were members of the class. If the printer or class removed is also the system default destination, the system no longer has a default destination.

To remove a printer or class, enter the following command:

> **/usr/lib/lpadmin -x** *printername* or *classname*

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Remove

If all you want to do is remove a printer from a class but not delete the printer, enter the following command:

> **/usr/lib/lpadmin -p** *printername* **-r** *classname*

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Modify

Managing the printing load

Occasionally, you may need to stop accepting print requests for a printer or move print requests from one printer to another. There are various reasons for doing this, such as the following:

- The printer needs periodic maintenance.
- The printer is broken.
- The printer was removed.
- The configuration was changed so that the printer can be used differently.
- Too many large print requests are queued for one printer and should be evenly distributed.

If you are going to make a big change in the way a printer is used, such as stopping its ability to handle a certain form, changing the print wheels available for it, or disallowing some people from using it, print requests that are currently queued for printing on it must be moved or canceled. The print service attempts to find alternate printers, but only if the user does not care which printer is used. Such requests are not automatically moved; if you do not move them first, the print service cancels them.

If you decide that a printer is to be taken out of service, its configuration is to be changed, or it is too heavily loaded, you can move print requests from it and reject additional requests for it. Use the **lpmove** and **reject** commands for this. If you do reject requests for a printer, you can later accept requests using the **accept** command.

Rejecting requests for a printer or class

To stop accepting any new requests for a printer or a class of printers, enter the following command:

> **/usr/lib/reject -r** *"reason" printername* or *classname*

Δ **sysadmsh** users select: Printers ⇨ Schedule ⇨ Reject

You can reject requests for several printers or classes in one command by listing their names on the same line, separating the names with spaces. The ***reason*** is displayed whenever anyone tries to print a file on the printer. You can omit this option if you do not want to give a reason.

Although the **reject** command stops any new print requests from being accepted, it does not move or cancel any requests currently queued for the printer. These continue to print as long as the printer is enabled.

Accepting requests for a printer or class

The **accept** command allows printers or classes of printers to accept print requests made with the **lp** command. You can allow a printer to accept requests after it has been properly configured.

After the condition that led to denying requests is corrected or changed, enter the following command to start accepting new requests:

> **/usr/lib/accept** ***printername*** or ***classname***

Δ **sysadmsh** users select: Printers ➪ Schedule ➪ Accept

Again, you can accept requests for several printers or classes in one command by listing their names on the same line. You will always have to use the **accept** command for a new printer or class after you have added it because the print service does not initially accept requests for new printers or classes.

Moving requests to another printer

If you have to move requests from one printer or class to another, enter one of the following commands:

> **/usr/lib/lpmove** ***request-id printername***
> **/usr/lib/lpmove** ***printername1 printername2***

Δ **sysadmsh** users select: Printers ➪ Request ➪ Move

You can give more than one request ID before the printer name in the first command. The first command moves the listed requests to the named printer. The latter command moves all requests currently queued for the first printer to the second printer. When the latter command is used, the print service also no longer accepts requests for the first printer (this has the same effect as the **reject** command).

lpmove, accept and reject examples

Here are some examples of how you might use **lpmove, accept**, and **reject**:

Example 1
You decide to change the ribbon on printer **printerl** and perform some preventive maintenance. You want to move all the requests for printer **printerl** to printer **printer2**. After the requests are moved, the print service no longer accepts requests for **printerl** (this has the same effect as a **reject printerl** command issued after the **lpmove** command).

```
/usr/lib/lpmove printerl printer2
```

Now you can disable the printer and start working on it. When you are finished, you can bring it back into service with the following command:

```
/usr/lib/accept printerl
```

At this point, if you had disabled the printer you should re-enable it. See the "Enabling and disabling printers" section in this chapter.

Example 2
You notice that someone has queued several large files for printing on the printer **laserl**. Meanwhile **laser2** is currently idle because no one had queued requests for it. You'll move the two biggest requests, **laserl-23** and **laserl-46** to **laser2**, and you reject any new requests for **laserl** for the time being.

```
/usr/lib/lpmove laserl-23 laserl-46 laser2
/usr/lib/reject -r "too busy--will reopen later" laserl
```

Example 3
You want to prevent printing requests from being routed to printer4 because of repairs:

```
/usr/lib/reject -r "printer4 needs repair" printer4
```

A user who requests a file to be printed on printer4 receives the following message:

```
UX:lp: ERROR: Requests for destination "printer 4" aren't
              being accepted

      TO FIX: Use the "lpstat -a" command to see
                why this destination is not accepting requests
```

To find out the acceptance status of printing destinations, enter:

lpstat -a

The output looks like this:

```
printername accepting requests since date time
printer4 not accepting requests since date time
    printer4 needs repair
```

Managing queue priorities

The print service provides a simple priority mechanism that people can use to adjust the position of a print request in the queue. Each print request can be given a priority level by the person who submits it; this is a number from 0 to 39, with *lower* numbers indicating *higher* levels of priority. Requests with higher priority (smaller numbers) are placed ahead of requests with lower priority (larger numbers).

In this way, if you decide that your print request is of too low a priority, you can set a higher priority (lower value) when you submit the file for printing. If you decide that your print request is of too high a priority, you can set a lower priority (higher value) when you submit the file for printing.

A priority scheme this simple does not work if there are no controls on how high one can set the priority. You can define the following characteristics of this scheme:

- Each user can be assigned a priority limit. One cannot submit a print request with a priority higher than his or her limit, although one can submit a request with a lower priority.
- A default priority limit can be assigned for the balance of users not assigned a personal limit.
- A default priority can be set. This is the priority given to print requests to which the user does not assign a priority.

By setting the characteristics according to your needs, you can prevent lower priority printing tasks (such as regular printing by most staff members) from interfering with higher priority printing tasks (such as payroll check printing by the accounting staff).

You may find that you want a critical print request to print ahead of any others, perhaps even if it has to preempt the currently printing request. You can have the print service give *immediate* handling to a print request and put on *hold* another print request. This lets the urgent print request print and delays the current print request until you have it *resumed.*

The **lpusers** command lets you assign both priority limits for users and priority defaults. In addition, you can use the **lp -i** ***request-id*** **-H hold** and **lp -i** ***request-id*** **-H immediate** commands to put a request on hold or move it up for immediate printing, respectively. These commands are discussed in detail in the sections that follow.

Setting priority limits

To set a user's priority limit, enter the following command:

> **/usr/lib/lpusers -q** ***priority-level*** **-u** ***username***

You can set the limit for a group of users by listing their names after the **-u** option. Separate multiple names with a comma or space (enclose the list in quotes if you use a space). The ***priority-level*** is a number from 0 to 39. As mentioned before, the lower the number, the higher the priority, or, in this case, the priority limit.

If you want to set the priority limit for all other users, enter the following command:

> **/usr/lib/lpusers -d** ***priority-level***

Δ **sysadmsh** users select: Printers ⇨ Priorities ⇨ Default

This sets the default limit; the default applies to those people who have not been given a personal limit using the earlier **lpusers** command.

If you later decide that someone should have a different priority limit, just re-enter the first command above with a new limit. If you decide that someone with a personal limit should have whatever the default limit is, enter the following command:

> **/usr/lib/lpusers -u** ***username***

Δ **sysadmsh** users select: Printers ⇨ Priorities ⇨ Remove

Again, you can do this for more than one person at a time by giving a list of names. Using the **lpusers** command with just the **-u** option puts the users in the *default limit* category.

If you do not set a default limit, people without personal limits are limited to priorities in the range of 20 to 39.

Setting a default priority

You can set the default priority that should be assigned to those print requests submitted without a priority. Use the following command:

/usr/lib/lpusers -q *priority-level*

Δ **sysadmsh** users select: Printers ➪ Priorities ➪ Highest

Do not confuse this default with the *default limit.* This default is applied when a user does not give a priority; the *default limit* is applied if you have not assigned a limit for a user – it is used to limit the user from giving too high a priority.

NOTE If the default priority is greater than the limit for a user, the limit is used instead.

If you do not set a default priority, the print service uses the default of 20.

Examining the priority limits and defaults

You can examine all the settings you have assigned for priority limits and defaults by entering the following command:

/usr/lib/lpusers -l

Δ **sysadmsh** users select: Printers ➪ Priorities ➪ List

Moving a request around in the queue

Once a user has submitted a print request, you can move it around in the queue to some degree. For example, you can:

- adjust the priority to any level regardless of the limit for the user
- put it on hold and let other requests print ahead of it
- put it at the head of the queue for immediate printing

You use the regular **lp** user command to do each of these.

Changing the priority for a request

Print requests that are still waiting to print can be reassigned a new priority. This repositions the request in the queue, putting it ahead of lower priority requests but behind any others at the same or higher priority. The priority limit assigned to the user (or the default priority limit) has no effect because you override this limit as the administrator.

Enter the following command to change the priority of a request:

lp **-i***request id* **-q** *new-priority-level*

You can change only one request at a time with this command. If a request is already printing, you cannot change its priority.

Putting a request on hold

Any request that has not finished printing can be put on hold. You can stop its printing, if it currently is printing, and keep it from printing until you resume it. Another user, however, cannot resume a print request that you put on hold.

Enter the following command to place a request on hold:

```
lp -i request-id -H hold
```

Enter the following command to resume the request:

```
lp -i request-id -H resume
```

Once resumed, a request continues to move up the queue and will print. If the request was printing when you held it, it is restarted and becomes the next request to print. Normally the request starts printing from the beginning, with page one, but you can have it start printing at a later page. Enter the following command to resume the request at a different page:

```
lp -i request-id -H resume -P starting-page-
```

The final dash is needed to specify the starting page and all subsequent pages.

NOTE The ability to print a subset of pages requires the presence of a filter that can handle this. The default filter used by the print service cannot handle it. An attempt to resume a request on a later page is rejected if an appropriate filter is not being used.

Moving a request to the head of the queue

You can move a print request to the head of the queue, where it is the next job eligible for printing. If it must start printing immediately, but another request is currently printing, you can hold the other request as described previously.

Enter the following command to move a print request to the head of the queue:

```
lp -i request-id -H immediate
```

Only the system administrator can move a request like this; regular users cannot use the **-H immediate** option.

NOTE If you set more than one request for immediate printing, they print in the reverse order set; that is, the request moved to the head of the queue most recently prints first.

Examining a printer configuration

Once you define a printer configuration, you probably want to review it to see if it is correct. If after examining the configuration you find you made a mistake, you can run the necessary command (or **sysadmsh** selection) for the characteristic concerned; you do not have to redefine the entire configuration.

Use the **lpstat** command to examine both the configuration and the current status of a printer. A short form of this command gives just the status; you can use it to see if the printer exists and if it is busy, idle, or disabled. A long form of the command adds the complete configuration. Enter one of the following commands to examine a printer:

lpstat -p ***printername***
lpstat -p ***printername*** **-l**

The second command is the long form. With either command you should see something like the following:

```
printer printer-name now printing request-id. enabled
since date.

printer printer-name is idle. enabled since date.

printer printer-name disabled since date.
       reason

printer printer-name waiting for auto-retry.
       reason
```

The "waiting for auto-retry" output shows that the print service failed in trying to use the printer (because of the ***reason*** shown) and that the print service will try again later.

With the long form of the command, you may also see the following items on the output:

```
Form mounted: form-name
Content types: content-type-list
Printer type: printer-type
Description: comment
Connection: connection-info
Interface: path-name
On fault: alert-method
After fault: fault-recovery
Users allowed:
        user-list
Forms allowed:
        form-list
Banner required
Character sets:
        character-set-list
Default pitch: integer CPI, integer LPI
Default page size: scaled-decimal-number wide,
   scaled-decimal-number long
Default port settings: stty-option-list
```

Using forms and filters

A preprinted form is a paper image of a blank form that you can load into your printer. An application typically generates a file that, when printed on the blank form, fills out the form. The print service includes facilities to create and administer forms.

The print service also permits the use of filters, which are used to accomplish three things:

- Convert a user's file into a data stream that prints properly on a given printer.
- Handle the various modes of printing that people may request with the **-y** option to the **lp** command, such as two-sided printing, landscape printing, draft or letter quality printing, and so on.
- Detect printer faults and inform the print service so that the latter can alert you.

There are very few cases where it will be necessary to use forms and filters, but these features are supported. For complete information, consult the **lpfilter**(ADM) and **lpforms**(ADM) manual pages.

Using the information in the request log

The directories */usr/spool/lp/temp* and *usr/spool/lp/requests* contain files that describe each request that has been submitted to the print service. Each request has two files, one in each directory, that contain information about the request. The information is split to put more sensitive information in the */usr/spool/lp/requests* directory where it can be kept secure. The request file in */usr/spool/lp/temp* is safe from all except the user who submitted the request, while the file in */usr/spool/lp/requests* is safe from even the submitting user.

These files remain in their directories only as long as the request is on the queue. Once the request is finished, the information in the files is combined and appended to the file */usr/spool/lp/logs/requests.*

The request log has a simple structure that makes it easy to extract data using common UNIX system shell commands. The requests are listed in the order in which they were printed and are separated by lines that give the request ID. Each line below the separator line is marked with a single letter that identifies the kind of information contained in the line. Each letter is separated from the data by a single space. Table 7.7 lists the log file codes.

Table 7-7 Request log entries

Letter	Content of line
"="	This is the separator line, containing the request ID, the user and group IDs of the user, the total number of bytes in the original (unfiltered) files, and the time when the request was queued. These items are separated by commas and are in the order just named. The user ID, group ID, and sizes are preceded by the word **uid, gid,** and **size,** respectively.
C	The number of copies printed.
D	The printer or class destination or the word **any.**
F	The name of the file printed. This line is repeated for each file printed, and files are printed in the order given.
f	The name of the form used.
H	The type of special handling used, spelled out (**resume, hold, immediate**). The only useful value found in this line is **immediate.**

(Continued on next page)

Table 7-7 Request log entries
(Continued)

Letter	Content of line
N	The type of alert used when the print request is successfully completed. The type is the letter **M** if the user was notified by mail, or **W** if the user was notified by a message to his or her terminal.
O	The **-o** options.
P	The priority of the print request.
p	The list of pages printed.
r	This single letter line is present if the user asked for raw processing of the files (the **-r** option of the **lp** command.)
S	The character set or print wheel used.
s	The outcome of the request as a combination of individual bits expressed in hexadecimal form. While several bits are used internally by the Spooler, the most important bits are listed below: **0x0004** Slow filtering finished successfully. **0x0010** Printing finished successfully. **0x0040** The request was canceled. **0x0100** The request failed filtering or printing.
T	The title placed on the banner page.
t	The type of content found in the file(s).
U	The name of the user who submitted the print request.
x	The slow filter used for the request.
Y	The list of special modes to give to the filters used to print the request.
y	The fast filter used for the request.
z	The printer used for the request. This differs from the destination (the **D** line) if the request was queued for any printer or a class of printers or if the request was moved to another destination by the print service administrator.

Customizing the print service

Although the print service tries to be flexible enough to handle most printers and printing needs, it cannot be complete. You may buy a printer that does not quite fit into the way the print service handles printers or may have a printing need that the standard features of the print service do not accommodate.

You can customize the print service in a few ways. This section tells you how you can:

- adjust the printer port characteristics
- adjust the *terminfo* database
- write an interface program

The diagram in Figure 7-2 gives an overview of the processing of a print request.

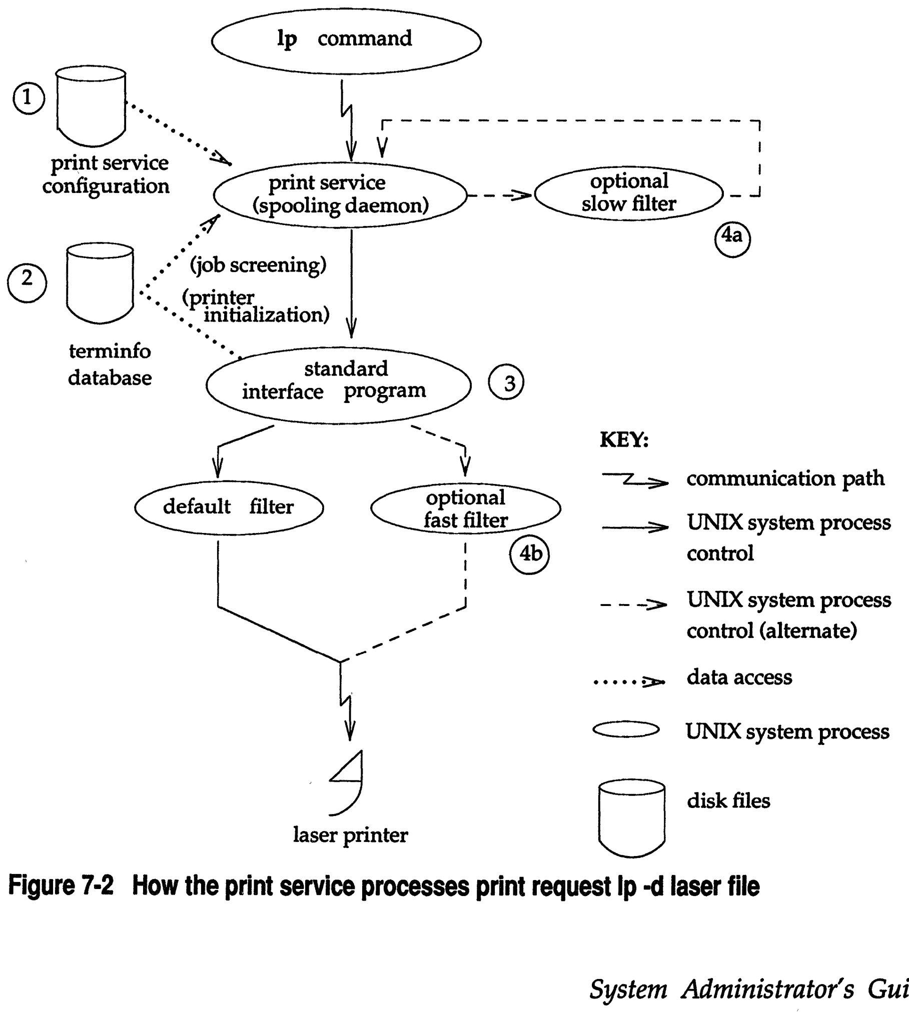


Figure 7-2 How the print service processes print request lp -d laser file

Each print request is sent to a *spooling daemon* (background program) that keeps track of all the requests. The daemon is created when you start the print service. This UNIX system process is also responsible for keeping track of the status of the printers and slow filters; when a printer finishes printing a user's file, the daemon starts it printing another request, if one is queued.

You can customize the print service by adjusting or replacing some of the pieces shown in Figure 7-2. (The numbers are keyed to the diagram.)

1. For most printers, you need only change the printer configuration stored on disk. The earlier sections of this chapter have explained how to do this. Some of the more printer-dependent configuration data are the printer port characteristics: baud rate, parity, and so on.
2. For printers that are not represented in the *terminfo* database, you can add a new entry that describes the capabilities of the printer. This database is used in two parallel capacities: screening print requests to ensure that those accepted can be handled by the desired printer and setting the printer so it is ready to print the request.

   For instance, if the *terminfo* database does not show a printer capable of setting a page length requested by a user, the spooling daemon rejects the request. On the other hand, if it does show it capable, then the same information is used by the interface program to initialize the printer.
3. For particularly difficult printers, or if you want to add features not provided by the delivered print service, you can change the standard interface program. This program is responsible for managing the printer: it prints the banner page, initializes the printer, and invokes a filter to send copies of the user's files to the printer.

4a,b. To provide a link between the applications used on your system and the printers, you can add slow and fast filters. Each type of filter can convert a file into another form, mapping one set of escape sequences into another, for instance, and can provide special setup by interpreting print modes requested by a user. Slow filters are run separately by the daemon to avoid tying up a printer. Fast filters are run so their output goes directly to the printer; thus, they can exert control over the printer.

Adjusting the printer port characteristics

You should make sure that the printer port characteristics set by the print service match the printer communication settings. The standard printer port settings were designed to work with typical UNIX files and many printers, but they do not work with all files and printers. This is not really a customizing step, because a standard feature of the print service is to allow you to specify the port settings for each printer. However, it is an important step in getting your printer to work with the print service, so it is described in more detail here.

When you add a new printer, read the documentation that comes with it so that you understand what it expects from the host (the print service). Then read the manual page for the **stty**(C) command. It summarizes the various characteristics that can be set on a terminal or printer port.

Only some of the characteristics listed in the **stty**(C) manual page are important for printers. The ones likely to be of interest to you are listed in the following table (but you should still consult the **stty**(C) manual page for others).

Printers connected directly to computers and those connected over some networks require that the printer port characteristics be set by the interface program. These characteristics define the low-level communications with the printer. Included are the baud rate; use of XON/XOFF flow control; 7, 8, or other bits per byte; style of parity; and output post-processing. The standard interface program uses the **stty** command to initialize the printer port, minimally setting the baud rate and a few other default characteristics.

The default characteristics applied by the standard interface program are listed in Table 7.8.

Table 7-8 Default stty options

Default	Meaning
9600	9600 baud rate
cs8	8-bit bytes
-cstopb	1 stop bit per byte
-parenb	no parity generation
ixon	enable XON/XOFF flow control
-ixany	allow only XON to restart output
opost	post-process data stream as listed below
-oluc	do not map lowercase to uppercase
onlcr	map linefeed into carriage-return/linefeed
-ocrnl	do not map carriage-return into linefeed
-nocr	output carriage-returns even at column 0
nl0	no delay after linefeeds
cr0	no delay after carriage-returns
tab0	no delay after tabs
bs0	no delay after backspaces
vt0	no delay after vertical tabs
ff0	no delay after formfeeds

You may find that the default characteristics are sufficient for your printers. However, printers vary enough that you are likely to find that you have to set different characteristics. See the **stty**(C) man page to find the complete list of characteristics.

If you have a printer that requires printer port characteristics other than those handled by the **stty** program, you must customize the interface program.

When you add a new printer, you can specify an additional list of port characteristics that should be applied when printing each user's file. The list you give will be applied after the default list so that you do not need to include default items that you do not want to change in your list. Specify the additional list as follows:

/usr/lib/lpadmin -p ***printer-name*** **-o "stty='*stty-option-list*'**

Δ **sysadmsh** users select: Printers ⇨ Configure ⇨ Parameters

Note that both the double quotes and single quotes are needed if you give more than one item in the ***stty-option-list***. If you do not include alternate printer port characteristics, the default list in the table will be used.

As one example, suppose your printer is to be used for printing graphical data, where linefeed characters should be output alone without an added carriage-return. You would enter the following command:

/usr/lib/lpadmin -p ***printer-name*** **-o "stty=-onlcr"**

Note that the single quotes are omitted because there is only one item in the list.

As another example, suppose your printer requires odd parity for data sent to it. You would enter the following command:

/usr/lib/lpadmin -p ***printer-name*** **-o "stty='parenb parodd cs7'"**

Adjusting the terminfo database

The print service relies on a standard interface and the *terminfo* database to initialize each printer and set up a selected page size, character pitch, line pitch, and character set. Thus, it is usually sufficient to have the correct entry in the *terminfo* database to add a new printer to the print service. Several entries for popular printers are delivered in *terminfo* database entries with the print service package.

Each printer is identified in the *terminfo* database with a short name; this kind of name is identical to the kind of name used to set the **TERM** shell variable. For instance, the AT&T model 455 printer is identified by the name **455**.

If you cannot find a *terminfo* entry for your printer, you should add one. If you do not, you may still be able to use the printer with the print service, but you cannot get automatic selection of page size, pitch, and character sets, and you may have trouble keeping the printer set in the correct modes for each print request. Another option to follow instead of updating the *terminfo* entry is to customize the interface program used with the printer. See the next section for details on how to do this.

There are hundreds of items that can be defined for each terminal or printer in the *terminfo* database. However, the print service uses less than fifty of these, and most printers need even less than that. Table 7.9 lists the items that need to be defined (as appropriate for the printer) to add a new printer to the print service.

Table 7-9 terminfo definitions

terminfo item	Meaning
Booleans:	
daisy	Printer needs operator to change character set
Numbers:	
bufsz	Number of bytes buffered before printing
cols	Number of columns in a line
it	Tabs initially every # spaces
lines	Number of lines on a page
orc	Horizontal resolution in units per character
orhi	Horizontal resolution in units per inch
orl	Vertical resolution in units per line
orvi	Vertical resolution in units per inch
cps	Average print rate in characters per second
Strings:	
cr	Carriage return
cpi	Change number of characters per inch
lpi	Change number of lines per inch
chr	Change horizontal resolution
cvr	Change vertical resolution
csnm	List of character set names
mgc	Clear all margins (top, bottom and sides)
hpa	Horizontal position absolute
cud1	Down one line
cuf1	Carriage right
swidm	Enable double wide printing
rwidm	Disable double wide printing
ff	Page eject
is1	Printer initialization string
is2	Printer initialization string
is3	Printer initialization string
if	Name of initialization file
iprog	Pathname of initializing program
cud	Move carriage down # lines
cuf	Move carriage right # columns
rep	Repeat a character # times
vpa	Vertical position absolute

(Continued on next page)

Table 7-9 terminfo definitions
(Continued)

terminfo item	Meaning
scs	Select character set
smgb	Set bottom margin at current line
smgbp	Set bottom margin
smgl	Set left margin at current column
smglp	Set left margin
smgr	Set right margin at current column
smgrp	Set right margin
smgt	Set top margin at current line
smgtp	Set top margin
scsd	Start definition of a character set
ht	Tab to next 8-space tab stop

Consult the manual page for the **terminfo**(M) file structure for details on how to construct a *terminfo* database entry for a new printer.

Once you make the new entry, you need to compile it into the database using the **tic** program. Just enter the following command:

```
tic filename
```

filename is the name of the file containing the *terminfo* entry you have crafted for the new printer.

NOTE The print service gains much efficiency by *caching* information from the *terminfo* database. If you add or delete *terminfo* entries or change the values that govern pitch settings, page width and length, or character sets, you should stop and restart the print service so it can read the new information.

How to write an interface program

NOTE If you have an interface program that you have used with the print service before UNIX System V Operating System Release 3.2, it should still work with the print service. Note, though, that several **-o** options have been standardized and are passed to every interface program. These may interfere with similarly named options your interface program uses.

If you have a printer that is not supported by simply adding an entry to the *terminfo* database, or if you have printing needs that are not supported by the standard interface program, you can furnish your own interface program. It is a good idea to start with the standard interface program (or one of the many others found in */usr/spool/lp/model*) and change it to fit, rather than starting from scratch. The "standard" script is found under the name */usr/spool/lp/model/standard.*

What does an interface program do?

Any interface program performs the following tasks:

- Initializes the printer port, if needed. The generic interface program uses the **stty** command to do this.
- Initializes the physical printer. The generic interface program uses the *terminfo* and the **TERM** shell variable to get the control sequences to do this.
- Prints a banner page, if needed.
- Prints the correct number of copies of the request content.

An interface program is not responsible for opening the printer port. This is done by the print service, which calls a dial-up printer if that is how the printer is connected. The printer port connection is given to the interface program as standard output, and the printer is set to be the controlling terminal for the interface program so that a hang-up of the port causes a SIGHUP signal to be sent to the interface program.

A customized interface program must not terminate the connection to the printer or in any fashion uninitialize the printer. This restriction allows the print service to use the interface program only for preparing the printer and printer port, while the printing of content is done elsewhere, by the print service, for example, for preprinted form alignment patterns.

How an interface program is used

When the print service routes an output request to a printer, the interface program for the printer is invoked as follows:

/usr/spool/lp/admins/lp/interfaces/*printer id user title copies options file1 file2 ...*

Arguments for the interface program are:

printer	Printer name. This is the name given to the interface program itself.
id	Request id returned by **lp**.
user	Login name of user who made the request.
title	Optional title specified by the user.
copies	Number of copies requested by user.
options	List of options separated by blanks, specified by user or set by the print service.
file	Full pathname of a file to be printed.

When the interface program is invoked, its standard input comes from */dev/null*, its standard output is directed to the printer port, and its standard error output is directed to a file that will be given to the user who submitted the print request.

The standard interface recognizes the following values in the list in *options*:

nobanner This option is used to skip the printing of a banner page; without it, a banner page is printed.

nofilebreak

This option is used to skip page breaks between separate data files; without it, a page break is made between each file in the content of a print request.

cpi=*decimal-number1*

lpi=*decimal-number2*

These options say to print with ***decimal-number1*** characters per inch and ***decimal-number2*** lines per inch, respectively. The standard interface program extracts from the *terminfo* database the control sequences needed to initialize the printer to handle the character and line pitches.

The words *pica, elite,* and *compressed* are acceptable replacements for the ***decimal-number1*** and are synonyms for *10* characters per inch, *12* characters per inch, and as many characters per inch as possible.

length=*decimal-number1*

width=*decimal-number2*

These options specify the length and width, respectively, of the pages to be printed. The standard interface program extracts from the *terminfo* database the control sequences needed to initialize the printer to handle the page length and page width.

stty=*'stty-option-list'*

The ***stty-option-list*** is applied after a default ***list*** as arguments to the **stty** command. The default list is used to establish a default port configuration; the additional list given to the interface program is used to change the configuration as needed.

The above options are either specified by the user when issuing a print request or by the print service from defaults given by the administrator for the printer (**cpi, lpi, length, width, stty**) or for the preprinted form used in the request (**cpi, lpi, length, width**).

Additional printer configuration information is passed to the interface program in shell variables:

TERM=*printer-type*

This shell variable specifies the type of printer. The value is used as a key for getting printer capability information from the extended *terminfo* database.

FILTER=*'pipeline'*

This shell variable specifies the filter to use to send the request content to the printer; the filter is given control of the printer.

CHARSET=*character-set*

This shell variable specifies the character set to be used when printing the content of a print request. The standard interface program extracts from the *terminfo* database the control sequences needed to select the character set.

A customized interface program should either ignore these options and shell variables or should recognize them and treat them in a consistent manner.

Customizing the interface program

Make sure that the custom interface program sets the proper stty modes (terminal characteristics such as baud rate or output options). The standard interface program does this, and you can follow suit. Look for the section that begins with the shell comment:

```
## Initialize the printer port
```

Follow the code used in the standard interface program. It sets both the default modes and the adjusted modes given by the print service or the user with a line like the following:

stty ***mode options*** **0<&1**

This command line takes the standard input for the **stty** command from the printer port. An example of an **stty** command line that sets the baud rate at 1200 and sets some of the option modes is shown here:

stty -parenb -parodd 1200 cs8 cread clocal ixon 0<&1

One printer port characteristic not set by the standard interface program is hardware flow control. This is set depending on your computer hardware. The code for the standard interface program suggests where this and other printer port characteristics can be set. Look for the section that begins with the shell comment

```
# Here you may want to add other port initialization code.
```

Because different printers have different numbers of columns, make sure the header and trailer for your interface program correspond to your printer. The standard interface program prints a banner that fits on an 80-column page (except for the user's title which may be longer). Look for the section in the code for the standard interface program that begins with the shell comment

```
## Print the banner page
```

The custom interface program should print all user-related error messages on the standard output or on the standard error. The messages sent to the standard error are mailed to the user; the messages printed on the standard output end up on the printed page, where they can be read by the user when they pick up the output.

When printing is complete, your interface program should exit with a code that tells the status of the print job. Exit codes are interpreted by the print service as shown in Table 7.10.

Table 7-10 Exit codes

Code	Meaning to the print service
0	The print request completed successfully. If a printer fault occurred, it was cleared.
1 to 127	A problem was encountered in printing this particular request (for example, too many non-printable characters or the request exceeds the printer capabilities). This problem does not affect future print requests. The print service notifies the person who submitted the request (via **write**(C) or **mail**(C)) that there was an error in printing it. If a printer fault occurred, it was cleared.
128	Reserved for internal use by the print service. Interface programs must not exit with this code.
129	A printer fault was encountered in printing the request. This problem affects future print requests. If the fault recovery for the printer directs the print service to wait for the administrator to fix the problem, it disables the printer. If the fault recovery is to continue printing, the print service does not disable the printer but tries printing again in a few minutes.
> 129	These codes are reserved for internal use by the print service. Interface programs must not exit with codes in this range.

As the table shows, one way of alerting the administrator to a printer fault is to exit with a code of 129. Unfortunately, if the interface program exits, the print service has no choice but to reprint the request from the beginning when the fault is cleared. Another way of getting an alert to the administrator but without requiring reprinting the entire request, is to have the interface program send a fault message to the print service but wait for the fault to clear. When the fault clears, the interface program can resume printing the user's file. When finished printing, it can give a zero exit code as if the fault never occurred. An added advantage is that the interface program can detect when the fault is cleared automatically so that the administrator does not have to enable the printer.

Fault messages can be sent to the print service using the **lp.tell** program. This is referenced using the **$LPTELL** shell variable in the standard interface code. The program takes its standard input and sends it to the print service, where it is put into the message that alerts the administrator to the printer fault. If its standard input is empty, **lp.tell** does not initiate an alert. Examine the standard interface code immediately after these comments for an example of how the **lp.tell ($LPTELL)** program is used:

```
# Here's where we set up the $LPTELL program to capture
# fault messages.

# Here's where we print the file.
```

With the special exit code 129 or the **lp.tell** program, there is no longer the need for the interface program to disable the printer itself. Your interface program can disable the printer directly, but doing so overrides the fault alerting mechanism. Alerts are sent only if the print service detects the printer has faulted, and the special exit code and the **lp.tell** program are its main detection tools.

If the print service has to interrupt the printing of a file at any time, it kills the interface program with a signal 15 (see **signal**(S) in the *Programmer's Reference*, and the **kill**(C) manual page).

If the interface program dies from receipt of any other signal, the print service assumes that future print requests are not affected and continues to use the printer. The print service notifies the person who submitted the request that it did not finish successfully.

The signals SIGHUP, SIGINT, SIGQUI, and SIGPIP (trap numbers 1, 2, 3, and 13) start out being ignored when the interface is invoked. The standard interface changes this to trap these signals at appropriate times. The standard interface considers receipt of these signals as meaning the printer has a problem and issues a fault. This is the program the print service uses to manage the printer each time a file is printed. It has four main tasks:

- to initialize the printer port (the connection between the computer and the printer),
- to initialize the printer (restore it to a normal state in case a previously printed file has left it in an unusual state) and set the character pitch, line pitch, page size, and character set requested by the user,
- to print a banner page, and
- to run a filter to print the file.

How to add an interface program

If you do not choose an interface program, the standard one provided with the print service is used. This should be sufficient for most of your printing needs. If you prefer, however, you can change it to suit your needs or completely rewrite your own interface program, and then specify it when you add a new printer.

If you plan to use the standard interface program, you need not specify it when adding a printer. However, if you use a different interface program, you can either refer to it by its full pathname or by another printer using the same interface program.

To identify a customized interface program by name, give the printer name and the pathname of the interface program as follows:

/usr/lib/lpadmin -p*printername* -i*pathname*

To identify a customized interface program by reference to another printer, give the printer names as follows:

/usr/lib/lpadmin -p*printername1* -e*printername2*

printername1 should be replaced with the name of the printer you are adding; ***printername2*** should be replaced with the name of the printer already added that is using the customized interface program.

To identify an interface program by reference to a model interface program, give the printer name and model name as follows:

> **/usr/lib/lpadmin -p** ***printername*** **-m*modelname***

This takes the interface program from the file */usr/spool/lp/model/modelname.*

NOTE If *printername* is a new printer to be added, the **lpadmin** command line must include the **-u** (dial-out) or **-v** (device) option. The printer must then be enabled and accepted.

Configuring a network printer

To configure the spooler to print to a printer on another machine accessed via network (such as TCP/IP or UUCP) you must set up the printer as follows:

1. Choose Printers ➪ Configure ➪ Add from **sysadmsh.** Give the printer a name, and choose the existing interface script network. The Connection should be "Direct" and the Device should be "Hardwired." If you are adding a printer on a TCP/IP network, use */dev/null* for the device name, otherwise, this should be left blank.
2. Create a file called */usr/spool/lp/remote.* This file should contain the following line:

   name:⟨Tab⟩*command*

   where ***name*** is the name of the printer on this machine and ***command*** is the command which will transfer the file to be printed to the remote system. For example, if the name you gave the printer is ***remprint*** and the name of the remote machine to which this printer is connected is ***wally*** and the name of the printer on machine ***wally*** is called ***beaver*** then the line would read:

   For UUCP:

   ```
   remprint: uux - wally!/usr/bin/lpr -dbeaver
   ```

   For TCP/IP:

   ```
   remprint: /usr/spool/lp/bin/rlpcmd wally lp -dbeaver
   ```

3. The file */usr/spool/lp/remote* should have owner and group **lp**, and be publicly readable. To set this, issue the following commands:

   chown lp /usr/spool/lp/remote
   chgrp lp /usr/spool/lp/remote
   chmod 444 /usr/spool/lp/remote

4. For a TCP/IP network, a user need not have an account on the print server; all that is necessary is consistency among the *lp* accounts on the network. The *.rhosts* file should contain entries in the following format:

   client `lp`

   All print jobs are submitted under the name *lp*.

NOTE If you have configured a remote printer over UUCP, some information about dialup printers also applies to UUCP. See the next section about dialup printers.

Configuring a dialup printer

To set up a dialup printer, you must do the following:

1. Because the **cu** program accesses a printer in the same way the print service does, you should set up the files as though preparing access to the printer for **cu**. The **cu** command is not used to access printers but can serve as a yardstick when setting up files: if **cu** can access a printer, the print service will be able to access it, too. (See the "Using modems" chapter in this guide for details about setting up a modem connection.)

   WARNING The *Devices* file entry you use for the dialup configuration should use a *Dialers* entry rather than a dialer binary.

2. Make certain the following line is present in */etc/default/lpd*:

   ```
   DIALUPPRINTER=YES
   ```

   If it is missing or the entry reads NO, change it.

3. Choose Printers ➪ Configure ➪ Add from **sysadmsh**. Complete the fields as follows:

Name:	dialup (or any name)
Interface:	dumb
Dial-up information:	the phone number of the remote system
Device:	hardwired

   Press ⟨Ctrl⟩x when you are finished.

4. Enter the following command to ensure failure info is mailed to *root*:

   ```
   /usr/lib/lpadmin -p dialup -A mail
   ```

5. You must then prepare the print system to accept jobs and enable the device:

   ```
   accept dialup
   enable dialup
   ```

6. You should then be able to submit job with the following command:

   lp -ddialup *filename*

The following applies to dialup and UUCP network printers: If the printer or port is busy, the print service will automatically retry later. This retry rate is 10 minutes if the printer is busy and 20 minutes if the port is busy. The rate is not adjustable. However, you can force an immediate retry by issuing an **enable** command for the printer. If the port or printer is likely to be busy for an extended period, you should issue a **disable** command.

The **lpstat -p** command reports the reason for a failed dial attempt. Also, if you are alerted to a dialing fault (see "Fault alerting" in this section), the alert message will give the reason for the fault. These messages are identical to the error messages produced by the UUCP system for similar problems. See the section called "UUCP STATUS error messages" in "Building a remote network with UUCP" in this guide for an explanation of the reasons for failure.

Setting up RTS/CTS protocol serial printers

The RTS and CTS lines for the RS-232 serial interface were originally intended as handshaking signals between a Data Terminal Equipment (DTE) device (computer, printer, and so forth) and a Data Communications Equipment (DCE) device (almost always a modem). The RTS (Ready To Send) line is asserted by the DTE when it is ready to send data to the DCE. The DCE asserts the CTS (Clear To Send) line when it was ready to receive data. If the CTS line goes low, then the DTE should stop sending data until CTS goes high again.

The UNIX system also uses the RTS line for handshaking in the other direction. If the printer sees that its input buffer is nearly full, it will lower the CTS line. The serial driver will then stop sending, and wait for the printer to catch up. The operating system will raise the CTS line when it is ready for more data.

Many printers use the DTR (Data Terminal Ready) line for handshaking rather than RTS or CTS. For these devices, the cable must be wired to connect the printer's DTR pin to the computer's CTS pin (see Figure 7-4).

To set up for RTS/CTS flow control, do the following:

1. Use the modem-control port (for example: */dev/tty1A*). If you plan to use the spooler to access this printer, make sure you specify the modem control port rather than one of the standard serial devices displayed when you use the **sysadmsh** Printers ➪ Configure ➪ Parameters selection asks you to enter a device name.
2. Make sure **stty** settings include **-ixon -ixoff -clocal rtsflow ctsflow.**
3. For a device that uses the RTS and/or CTS lines for handshaking, the cable should be wired as shown in Figure 7-3.

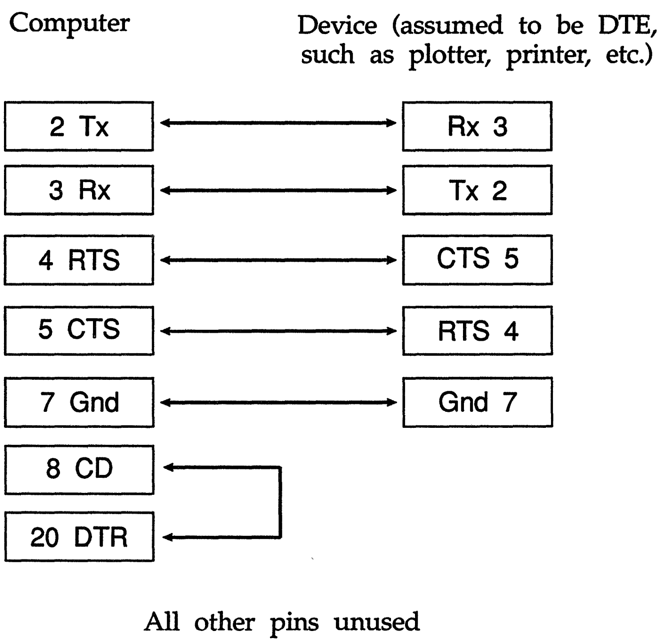


Figure 7-3 RTS/CTS handshaking

4. If the device uses the DTR line for handshaking, the cabling should be as shown in Figure 7-4.

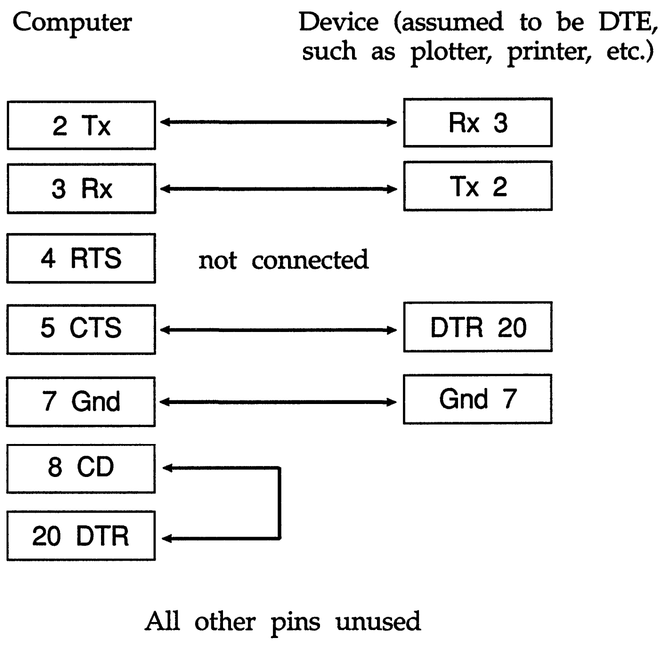


Figure 7-4 DTR handshaking

5. If the information contained here does not solve the problem, try removing **rtsflow** from the **stty** command string.

Using a printer without the spooler

If you use a printer without the spooler, any **stty** settings you have specified for use with that printer do not stay in effect. The spooler opens the file and then runs the **stty** commands as specified in the printer interface script. To use a printer without the spooler, follow the instructions in this section.

While logged in as *root*, give the following commands or insert them into the initialization file */etc/rc2.d/S80lp* before the line that calls */usr/lib/lpsched.* The first command is for serial printers, and the second for parallel printers:

```
(stty baud ixon ixoff -ixany ; cat > /dev/null) < /dev/ttyn &
(stty onlcr; while : ; do sleep 3600; done) < /dev/lpn &
```

where ***baud*** is the baud rate of the printer, tty*n* or lp*n* is the serial or parallel device name, respectively. This command sets the stty options and holds the port open for use without the spooler. If you ever need to enable the port, make sure you kill this process first.

NOTE With certain multiport cards, it is necessary to add a **sleep** command after the initialization program supplied with the card, ***initprogram,*** followed by the **stty** holdopen command:

```
initprogram &
sleep 3
```

Creating an init device file

The standard parallel printer devices (*/dev/lp, /dev/lp0, /dev/lp1,* and */dev/lp2*) send a printer initialization string (**init**) the first time the device is opened after the system starts up. This is done on the first open only, so printers with large text buffers are not flushed by the sending of another file.

Some parallel printers require initialization every time a file is received for printing. Others require an **init** if the printer is turned off and back on again (for example, after changing paper or ribbons). The symptom of this situation is that the printer works fine until it is turned off and then back on.

If you need to initialize the printer more often than the standard devices provide, you can create an additional device file for the parallel port in use. This init device file can be used when necessary to initialize the printer.

1. Log in as super user.
2. Determine which device is the parallel port you are using. This example assumes the device is the main parallel port (*/dev/lp0*).
3. Run the command:

   ```
   cd/; fixperm -csdLPR /usr/lib/mkdev/perms/LPR
   ```

   to correct/initialize file permissions for the printer devices file */usr/lib/mkdev/perms/LPR.*
4. Use the **sysadmsh** selection:

   Printers ⇨ Configure ⇨ Modify

   to associate one of the parallel **init** devices (*lp0i, lp1i, lp2i*) with the printer.

   If your printer needs an **init** when it is turned off and on, use the following command line after the printer is turned on. Initialize the printer before the first file is sent to the printer (this example assumes the main parallel port):

   ```
   >/dev/lp0i
   ```

   If your printer needs an **init** every time a file is sent (and it does not have a large internal text buffer), you can use the */dev/lp0i* device all the time.

The **lp**(C) command will then send an **init** every time a file is sent to the printer.

Chapter 8

Using floppy disks and tape drives

An important part of any computer system is the ability to offload files and restore them when needed. There are several types of media used to store and recall files. Among these are floppy disks and magnetic tape devices. This chapter explains how to install and use storage media with your system. Your system should come with at least a floppy disk drive already installed and ready to run. This chapter provides instructions on how to add tape drives and how to use floppy disks.

Using tape drives

A tape drive is a mass storage device that uses magnetic tape cartridges to store data. A tape can hold many times the data that can be stored on floppies, making it much more useful for large backup operations.

The drives that are supported are listed in the *Release Notes*. For hardware-specific information, refer to the manual for your drive and the **tape**(HW) manual page.

Installation and configuration

Read your tape drive hardware manual for physical installation instructions and general information.

To add a tape drive, log in as *root* and enter the following:

mkdev tape

Δ **sysadmsh** users select: System ➪ Hardware ➪ Tape

The following menu is displayed:

```
Tape Drive Configuration Program

         1. Install a Tape Drive
         2. Remove a Tape Drive
         3. Change default Tape Drive

         Select an option or enter q to quit:
```

Enter **1** to add the drive. You are then asked to select the type of tape drive you have installed:

```
Tape Drive Installation Menu

         1. Install Cartridge Tape Drive
         2. Install Mini-Cartridge Tape Drive
         3. Install Qic-40 or Qic-80 Tape Drive
         4. Install SCSI Tape Drive

Select an option or enter q to return to the main menu:
```

The subsections that follow describe the configuration requirements for each drive type. Be sure and consult the sections on "Kernel relinking" and "Boot messages and /etc/default/boot" after following the instructions for your drive type.

NOTE AT-type hardware uses jumpers or switches to configure such characteristics as interrupt vector or DMA channel. EISA and Micro Channel hardware settings are configured with a Reference (setup) diskette provided with your machine. When you see instructions to set jumpers or switches, Micro Channel and EISA installers should use this information in conjunction with their reference or setup diskette.

Cartridge tape

The Cartridge Tape Driver selection refers to the QIC-02-type full-size cartridge tape drives. Follow these steps:

1. When you select the Cartridge Tape, another menu is displayed that controls the tape parameters:

```
QIC Cartridge Tape Configuration

1) Display current tape parameters
2) Modify current tape parameters
3) Select previous tape parameters
4) Select default tape parameters

Enter your choice or q to quit:
```

   The system has default values for each supported drive; these values are documented in Appendix A of the *Release Notes.* You need not modify these defaults unless you have configured your tape drive differently. Enter **q** and press ⟨Return⟩ if you are using the default values and skip to step 3.

2. If you have a non-default configuration, you must enter **2** to modify the default parameters. You see a display similar to the following:

```
  Tape Parameters      Values     Comments
  ---------------      ------     --------

1. Controller Type       0        1 = archive    3 = wangtek  4 = emerald
                                  5 = mountain   6 = tecmar   7 = everex
2. DMA Channel           0        1 or 3
3. Interrupt Vector      0        logical vector number
4. Base Address          0H       i/o addresses start here

Enter a parameter to modify or 'q' to return to the main menu.
```

   Select each parameter you wish to change and supply a value for each of the four categories listed above. Explanations of each category follow:

Controller type	This is the kind of tape drive/controller you have installed. The values and manufacturer are shown in the previous display.

DMA channel This is the Direct Memory Access (DMA) Channel that is being used. The choices are DMA Channel 1 or DMA Channel 3. On a typical tape controller card, you have a set of jumper pins for DACK and another set for DRQ. Each of these should have three pairs of pins labeled 1, 2, and 3. The DACK and DRQ pins should each be jumpered for the appropriate value (1 or 3) as they combine to give you your DMA value. In other words, if you want a DMA value of 3, you should jumper DACK for 3 and DRQ for 3. Either DMA 1 or 3 are acceptable and neither provides any particular advantage over the other.

Interrupt vector This is the interrupt value being used. On a tape controller card, there is an IRQ set of jumper pins, typically labeled 2-7. You can use any interrupt you choose, as long as it is not being used by another device on your system. Standard interrupts for ISA machines are shown in Table 8.1.

Table 8-1 Typical device interrupts (ISA)

Interrupt	Device
0*	Clock
1*	Console
2	Networks, Tapes and others
3	Serial COM2
4	Serial COM1
5	Alternate Parallel Port (lp2)
6*	Floppy Disk
7	Main Parallel Port (lp0 or lp1)

Never use interrupts 0, 1, or 6, as they are always used for standard devices on your system. The rest are potentially available, unless devices that typically use them are installed on your system. For example, if you have a COM1 serial port on your system and it is recognized on the boot screen, then interrupt 4 is not an available interrupt, unless you disable your COM1 port. On ISA machines, if a device is recognized by the UNIX system on the boot screen, the interrupt it uses is taken even if you are not actively using the device. In this case, you must either use another free interrupt, or physically disable the device using the interrupt you want to use for the tape controller.

The most commonly available interrupts are 2 and 5, as most systems do not come standard with an LP2 port or a network card. Be aware that if you do choose to use interrupt 2, you need to enter a value of 9. This is the only case where a value other than the actual number is given.

When you determine the interrupt you will be using, jumper the appropriate IRQ pins on your tape controller card and then indicate the same value (except in the case of interrupt 2 where you should indicate 9).

Base address This is the I/O address that your tape controller card is configured for. A typical tape controller card will have either jumper pins or switches for configuring the address. Check your tape hardware documentation to determine what the available address settings are. The system does not really care what address you are using, as long as it is not being used by another device on your system. This is not usually an issue, as common tape controller addresses are not often used by other devices.

Be sure and enter the address with a preceding **0x**. For example, if your address is 300 (hexadecimal), you should enter **0x300**.

3. When you have finished setting your parameters, you are returned to the tape parameter menu. You must then enter **q** and press ⟨Return⟩.
4. The system configuration is updated and the new device files are created and displayed on the screen. Table 8.4 lists the device files for each drive type and their purpose. You are also prompted for whether you wish to change the boot string as described in "Boot messages and /etc/default/boot" later in this chapter.
5. You are then returned to the tape configuration menu. Depending on whether or not the necessary driver is already present in the kernel, you may also be asked to relink the kernel as described in the section on "Kernel relinking."

Mini-cartridge (Irwin)

Mini-tape drives use the floppy disk drive controller and are significantly different from standard QIC tape drives. These units are also known as "Irwin" tape units. They are not configurable and do not require parameters to be entered. In addition, mini-cartridge tapes must be formatted before they can be used (preformatted tapes are strongly recommended; see "Tape formatting" later in this chapter). Make certain your drive is correctly jumpered; this can vary with different brands of machines. Refer to your hardware documentation and the *Release Notes* for more information.

The prompts for a mini-cartridge (Irwin) drive are as follows:

1. First you see:

   ```
   If you have changed the unit number for the Mini-Cartridge Tape Drive
   enter the new number [1-n] or press <Return> to use the default
   value of 1 or enter q or return to the main menu:
   ```

   Unless you are configuring the drive at a different unit number, enter **1** and press ⟨Return⟩.

2. The system configuration is updated and the new device files are created and displayed on the screen. Table 8.4 lists the device files for each drive type and their purpose. You are also prompted for whether you wish to change the boot string as described in "Boot messages and /etc/default/boot" later in this chapter.

3. You are then returned to the tape configuration menu. Depending on whether or not the necessary driver is already present in the kernel, you may also be asked to relink the kernel as described in the section on "Kernel relinking."

4. The Irwin driver also includes a configuration file */etc/default/mcconfig* that can contain options for debugging, hardware-specific options, and other useful features. The *mcconfig* file does not exist by default, it must be created and entries added. Table 8.2 contains some commonly used *mcconfig* entries.

Table 8-2 /etc/default/mcconfig entries

String	Meaning
`iropt=F`	Floating drive search (Adaptec controller)
`iropt=M`	Micro Channel bus
`iropt=I`	Wait-for-index before transfer of each block
`4100=P:370,I:6,D:2, T:2,T:0`	Specifies 4100 PC Bus Controller parameters

Refer to the **mcconfig**(F) manual page for more information on these and other options.

In addition, the Irwin driver includes a daemon process */etc/mcdaemon* that is automatically started at boot time. An Irwin drive will not function in single-user mode unless the daemon is started. To start the daemon while in single-user mode, enter the following command:

/etc/mcdaemon

QIC-40/80

These units are specialized mini-cartridge units that use a different format from the Irwin units described earlier. Refer to your tape drive documentation for instructions on how to jumper your drive properly. Soft Select-compatible drives do not use jumpers.

The prompts are as follows:

1. You are first asked to select the drive type, QIC-40 or QIC-80:

```
1. Install a Qic-40 Tape Drive
2. Install a Qic-80 Tape Drive

Select an option or enter q to return to the main menu:
```

   Select the appropriate option. If you are unsure of your choice, refer to the manual for your tape drive.

2. You are asked if you wish to enable extended tape length mode:

```
1. Do not enable Extended Length Mode
2. Enable Extended Length Mode

Select an option or enter q to return to the main menu:
```

   Select the appropriate option; you should not enable extended length mode if your drive does not support it. Refer to the **tape**(HW) manual page for information on extended length mode.

3. You see:

```
Enter one of the following:

     1-4          the drive select unit number for the
                  Qic-40 or Qic-80 Tape Drive

     s            a Soft Select mode drive with no select
                  jumpers installed

     Return       the default value of 2

Select an option or enter q to return to the main menu:
```

   If your drive supports Soft Select, enter **s** and press ⟨Return⟩. If your drive is jumpered at the default value of 2, simply press ⟨Return⟩. Otherwise, enter the appropriate number and press ⟨Return⟩. Refer to your hardware manual to determine your drive's current drive select option setting, or if your drive is in Soft Select mode. There are other configuration options available that control memory and buffer usage; see **tape**(HW) for details.

4. The system configuration is updated and the new device files are created and displayed on the screen. Table 8.4 lists the device files for each drive type and their purpose. You are also prompted for whether you wish to change the boot string as described in "Boot messages and /etc/default/boot" later in this chapter.

5. You are then returned to the tape configuration menu. Depending on whether or not the necessary driver is already present in the kernel, you may also be asked to relink the kernel as described in the section on "Kernel relinking."

SCSI

SCSI tape drives are attached to a SCSI adapter. You are asked to provide the device ID number (0-7), the number of the adapter the drive is attached to, and the logical unit number of the device (LUN). See Figure 8-1.

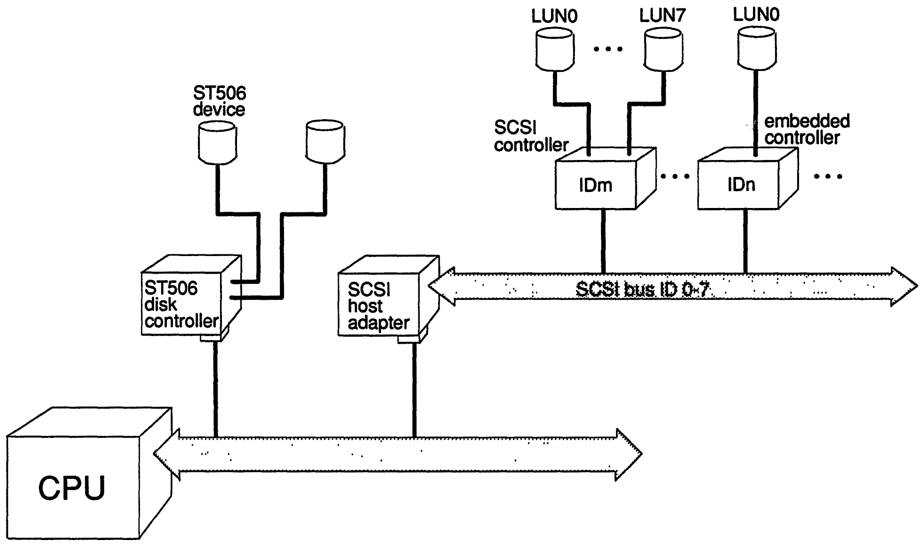


Figure 8-1 SCSI configuration schematic

As shown, a SCSI host adapter (HA) translates signals from the CPU bus to the SCSI bus. A SCSI controller is known as a "SCSI ID". A SCSI device is referenced by a logical unit number (LUN).

To configure a SCSI tape device, you must know:

- ID number of the controller (0-7) on the host adapter; the host adapter itself is usually ID 7, giving it the highest priority on the SCSI bus
- host adapter number (0 or 1)
- host adapter type; this code refers to the type of host adapter. Several types are supported. For example: **ad** refers to an Adaptec host adapter (**ad** is also the default type).

Because the tape drive and its controller are one unit (referred to as "embedded") the LUN or logical unit number is simply 0.

The **mkdev tape** prompts for SCSI configuration are as follows:

1. You see this menu:

```
SCSI Tape Drive Configuration Menu

          1. Install SCSI Cartridge Tape Drive
          2. Install SCSI Exabyte Tape Drive
          3. Install SCSI 9-track Tape Drive
          4. Install SCSI DAT Tape Drive
          5. Install Compaq SCSI Tape Drive

          Select an option or enter q to return to the main menu
```

   Enter the number associated with your tape drive and press ⟨Return⟩. If you are configuring a Compaq SCSI tape drive, proceed to the next section "Compaq SCSI."

2. You see the following message:

```
The type Tape Drive must be configured before use.
Do you wish to configure the type Tape Drive now? (y/n)
```

   Enter y and press ⟨Return⟩.

3. Now you must identify the type of adapter that the drive is attached to:

```
Enter the prefix of the SCSI host adapter that supports this device
or press ⟨Return⟩ for the default: type
Enter h for a list of host adapters or enter q to quit:
```

   Enter **h** to display a list of host adapters, and select one of them.

4. Now you are prompted for the number of the adapter:

```
Which 'hatype' SCSI host adapter supports this device?
Select 0-7 or q to quit:
```

   Enter **0** if it is the first host adapter, **1** for the second host adapter, and so forth.

5. If you are adding a device to a previously configured host adapter, or to a host adapter that is the only one of its kind installed in your system, skip to step 8. If you are configuring a second host adapter of a type already installed, you see the following:

```
hatype SCSI host adapter ha is not configured.
Would you like to update the link-kit?
```

The kernel already recognizes one of each type of host adapter; an additional entry must be created for the new one. Respond **y** and press ⟨Return⟩.

6. You may be prompted for one or more of the following, depending on your host adapter type (the system attempts to autoconfigure any values it can obtain from default files and other adapters installed of the same type):

```
What is the interrupt vector for this adapter?

What is the start IO address (hex) for this adapter?

What is the end IO address (hex) for this adapter?

What is the start controller memory address (hex) for this adapter?

What is the end controller memory address (hex) for this adapter?
```

The hardware appendix of the *Release Notes* contains a table of default SCSI host adapter configuration parameters. If you do not have a given type of adapter installed, you can probably use the values for another adapter. (You should check for conflicts before entering values; other hardware devices may already be using the same interrupt vector, I/O address, and so forth.) Enter each value as prompted and press ⟨Return⟩.

7. You are asked to confirm the values you supplied. Then the following prompt is displayed:

```
The following parameters will be used to configure hatype SCSI host adapter ha

list of parameters

Update the link-kit?
```

If you need to make corrections, enter **n** and press ⟨Return⟩, and then press ⟨Del⟩ to exit **mkdev tape** and start again. If the values are correct, enter **y** and press ⟨Return⟩ to add these values to the link kit.

8. Next, you see:

```
What is the Target ID for this device?
Select 0-7, or 'h' for help, or 'q' to quit:
```

   Enter the number of the controller attached to the adapter.

9. You are then prompted:

```
What is the LUN of this device?  Press <Return> to use the default: 0
Select 0-7, or 'h' for help, or 'q' to quit:
```

   Enter the number of the device attached to the controller. With most devices, the controller and the device are a single unit, in which case the Logical Unit Number is 0. Non-embedded controllers can support up to eight SCSI devices. In this case, LUN numbers are determined by jumper settings on each device. The valid range of LUN numbers is 0-7.

10. The information you supplied earlier for the device configuration is then displayed as in the following example. You are asked to confirm that you wish to update the configuration:

```
You are about to add the following SCSI device:

Host
Adapter          Adapter
Type    Device  Number  ID      LUN
-----------------------------------
-        -        -       -       -

Update SCSI configuration (y/n)?
```

   If the information is correct, enter y and press ⟨Return⟩.

11. The system configuration is updated and the new device files are created and displayed on the screen. Table 8.4 lists the device files for each drive type and their purpose. You are also prompted for whether you wish to change the boot string as described in "Boot messages and /etc/default/boot" later in this chapter.

12. You are then returned to the tape configuration menu. Depending on whether or not the necessary driver is already present in the kernel, you may also be asked to relink the kernel as described in the section on "Kernel relinking."

Compaq SCSI

The prompts for Compaq SCSI tape installation are as follows:

1. First you see:

```
Enter the SCSI ID of the tape being added or press <Return> for default (0):
```

   Enter the number of the SCSI host adapter to which the tape drive is connected, or acknowledge the default by pressing ⟨Return⟩.

2. The following menu is displayed:

```
Compaq SCSI Host Adapter Configuration

         1. Display current parameters
         2. Modify current parameters
         3. Select default parameters

Enter an option or 'q' to quit:
```

   Select **2** or **3** and press ⟨Return⟩.

3. You see the following:

```
Index  Parameters        Current Value   Valid Values
-----  ------------      -------------   ------------------------
  1    Base Address        130           130, 330 or *120 (Hex)
  2    DMA Channel           7           7 or 5
  3    Interrupt Vector      0           5, 3, *10, or *11

                         *Not supported SCSI Tape Adapter (001379)
```

   If you chose option 3, proceed to step 6. If you chose option 2, you also see the prompt:

```
Enter index of the parameter to modify or 'q' to return to the main menu:
```

   Enter 1, 2, or 3 to alter the base address, DMA channel, and interrupt vector, as desired. The display is repeated each time a parameter is altered. Enter **q** when the values are correct.

4. When the "Compaq Host Adapter Configuration" menu is displayed, enter **q** and press ⟨Return⟩ to continue.

5. The system configuration is updated and the new device files are created and displayed on the screen. Table 8.4 lists the device files for each drive type and their purpose. You are also prompted for whether you wish to change the boot string as described in "Boot messages and /etc/default/boot" later in this chapter.

6. You are then returned to the tape configuration menu. Depending on whether or not the necessary driver is already present in the kernel, you may also be asked to relink the kernel as described in the section on "Kernel relinking."

Device files

After the drive is configured, a series of device files are created to access the tape drive. Some are linked to the basic tape device file, */dev/rct0*. Others include the "no-rewind" device, which does not rewind the tape after access, and the "no-unload" device, which does not eject the tape after use.

Table 8.3 lists the prefixes used in tape device names.

Table 8-3 Tape device prefix

Prefix	Meaning	Example
e	ECC device (QIC-02 only)	/dev/erct0
rh	high density (9-track only)	/dev/rhStp0
nr	no rewind, no unload	/dev/nrStp0
nur	rewind, no unload	/dev/nurStp0
ur	rewind, unload	/dev/urStp0
r	rewind on close	/dev/rStp0
x	override/control	/dev/xStp0
n	no rewind	/dev/nrct0

Table 8.4 lists the main devices created for tape drives and the default device to which they are linked.

Table 8-4 Tape device files

Tape device	Device file	Default device
QIC-02	/dev/rct0	/dev/rct0
Mini (irwin)	/dev/rmc0	/dev/rctmini
QIC-40/80	/dev/rft0	/dev/rctmini
SCSI cartridge	/dev/urStp0	/dev/rct0
DAT/Exabyte	/dev/nurStp0*	/dev/rct0
9-track	/dev/nurStp0	/dev/rct0
Compaq	/dev/rmt/cst2*	/dev/rct0

In addition, some devices are linked to equivalent standard UNIX devices in the */dev/rmt/** hierarchy.

* Some drives have additional device files that depend on brand and model. For example, certain DAT drives support tape partitioning; see **dat**(HW) and **tape**(HW).

Kernel relinking

If it is necessary to relink the kernel, you see several prompts after exiting the tape drive installation menu. You are given the option of not relinking in case you are adding a number of devices. This way the kernel needs to be relinked only once.

1. First you see:

   ```
   You must create a new kernel to effect the driver change you specified.
   Do you wish to create a new kernel now? (y/n)
   ```

   Answer y to add the driver to your kernel.

2. Next, you see:

   ```
        The UNIX operating system will now be rebuilt.
        This will take a few minutes.  Please wait.

        Root for this system build is /.
   ```

   As part of the linking process, you see the following messages:

   ```
        The UNIX kernel has been rebuilt.

   Do you want this kernel to boot by default? (y/n)
   ```

   Answer y if you want this kernel to be used every time you boot the system.

3. The following is displayed:

   ```
   Backing up /unix to /unix.old.
   Installing new /unix.

   The kernel environment includes device node files and /etc/inittab.
   The new kernel may require changes to /etc/inittab or device nodes.

   Do you want the kernel environment rebuilt? (y/n)
   ```

   Enter y.

4. The following is displayed:

   ```
   The new kernel has been successfully linked and installed.
   To activate it, reboot your system.

   Setting up new kernel environment.
   ```

The tape drive unit is available for use after rebooting.

Boot messages and /etc/default/boot

When the kernel recognizes a tape drive (and when the driver is linked into the kernel) a message is always displayed at boot time indicating the device is present. This information can also be displayed using **hwconfig**(C). Table 8.5 contains the messages displayed, indicating the drive type.

Table 8-5 Tape drive boot messages

Type	Boot display message				
QIC-02*	`%tape`	`0x0338-0x033C`	`05`	`1`	`type=W`
mini	`%ctmini`	-	-	-	`type=ir`
QIC-40	`%ctmini`	-	-	-	`type=qic40`
QIC-80	`%ctmini`	-	-	-	`type=qic80`
SCSI*	`%tape`	-	-	-	`type=S ha=0 id=2 lun=0`

In addition, the default tape drive is contained in the boot string, a part of the boot command used when you press ⟨Return⟩ at the "Boot:" prompt. The tape boot string ensures that you can tell the system to recognize the tape drive even if you are not booting from the default kernel (as in the case when you are booting from the Emergency Boot Floppy and you need to recover the system from backup tapes). Here is an example of a tape bootstring for a Wangtek cartridge tape:

```
ct=wangtek(0x338,5,1)
```

Each time you install a new tape drive, you are prompted if you wish to change the default tape bootstring:

```
Enter new string, "rm" to remove string,
or enter q to leave current string as is:
```

You can change the string if desired, or enter **rm** to delete the existing string, or just enter **q** to finish the tape configuration.

Changing the default tape drive

If you install more than one tape drive, you are prompted by **mkdev tape** for the drive you want linked to the default tape device files. For example, if you have a both a SCSI cartridge tape drive and an 8-mm tape drive installed, only one of them can be linked to the default device */dev/rct0*. If you decide later to change the drive linked to the default device, you can do so with option 3 of the **mkdev tape** menu.

* The specifics of these configurations are examples.

Setting the default tar(C) device

After you install your tape drive, you must enter the correct size setting in the */etc/default/tar* file. When you edit the file, you see several entries for various default devices. Figure 8-2 shows the */etc/default/tar* file provided with the 96tpi distribution. Note that the sizes for */dev/rct0* and */dev/rctmini* are 0, meaning there is no size associated with this device. This is acceptable if you never use an entire tape. However, if you exceed the tape's capacity, **tar** cannot create a multi-volume archive properly. If you plan to create archives that span tape volumes, you must add the proper tape size as described in this section.

```
# device                     block   size   tape
archive0=/dev/rfd048ds9      18      360    n
archive1=/dev/rfd148ds9      18      360    n
archive2=/dev/rfd096ds15     10      1200   n
archive3=/dev/rfd196ds15     10      1200   n
archive4=/dev/rfd096ds9      18      720    n
archive5=/dev/rfd196ds9      18      720    n
archive6=/dev/rfd0135ds18    18      1440   n
archive7=/dev/rfd1135ds18    18      1440   n
archive8=/dev/rct0           20      0      y
archive9=/dev/rctmini        20      0      y
#The default device...
archive=/dev/rfd096ds15      10      1200   n
```

Figure 8-2 /etc/default/tar file

The utilities **xbackup**(ADM) and **xrestore**(ADM) have similar files and entries. For more information on default files, see the **default**(F) manual page and the manual entry for the particular backup or restore command.

QIC cartridge drives

The */dev/rct0* entry accesses the QIC cartridge tape drive. The cartridge sizes are indicated in Table 8.6. You can edit */etc/default/tar* and set the appropriate size, or create multiple entries to accommodate different tape sizes.

Table 8-6 QIC cartridge sizes

Length in feet	Entry in 'Size' field	Capacity in megabtyes
300	30000	30
450	45000	45
600	60000	60
1200	120000	120
2500	250000	250

Mini-cartridge (Irwin) drives

Find the entry in your */etc/default/tar* file for */dev/rctmini*. In Figure 8-2, this is **archive9**. The correct size for your *rctmini* device varies with the size of the tape you use. See Table 8.7.

Table 8-7 Mini-cartridge sizes

Tape size in megabytes	Actual capacity in megabytes	Entry in 'Size' field
10	8	8000
20	17	17000
40	35	35000
80	72	72000

The actual size is smaller because some storage is reserved for error correction code (ECC) data.

QIC-40/80 mini-cartridge drives

If you wish to use the default file with a QIC-40/80 drive, you must change one of the entries to use the */dev/rft0* device:

```
archive9=/dev/rft0      20      size    y
```

Replace *size* with the size (in Kbytes) of the cartridges you are using.

DAT and 8-mm (Exabyte) drives

If you wish to use the default file with a DAT or 8-mm drive, you must change one of the entries to use the */dev/rStp0* device:

```
archive9=/dev/rStp0     20      size    y
```

Replace *size* with the size (in Kbytes) of the cartridges you are using.

Archiving files on tape

You use a tape drive much like a floppy, but the volume of data stored is much greater. Tapes are much better for storing (backing up) entire filesystems. The **tar**(C) command is the recommended archive program for users and is best used for general archiving or transporting of files. Other programs such as **backup**(ADM) and **restore**(ADM) are meant for system administrators making copies of entire filesystems. Consult "Backing up filesystems" in this guide for making regular backups of filesystems.

The **cpio**(C) command is a general-purpose archive program that uses a different format than **tar**. The **dd**(C) program transfers or converts archives of unusual format; the input and output format can be specified on the command line.

The tar command

The **tar** command is useful for making a backup copy of entire directories. The command has the syntax:

```
tar cvf devicefile files
```

The ***devicefile*** is the filename that corresponds to the cartridge drive. ***files*** are the names of the files or directories to be copied. For example, to copy all the files in the directory */u/bogart* to the cartridge drive */dev/rct0*, enter:

```
tar cvf /dev/rct0 /u/bogart
```

Δ **sysadmsh** users select: Media ➪ Archive

Alternatively, you can use the shorter */etc/default/tar* entry (archive8) shown in Figure 8-2 rather than entering the full device name:

```
tar cv8 /u/bogart
```

To restore files stored on tape, insert the cartridge containing the files or directories you wish to restore and enter the following command:

```
tar xvf devicefile
```

Δ **sysadmsh** users select: Media ➪ Extract

Again, you can also use the shorter form:

```
tar xv8
```

tar restores all the files on the tape to the original directory.

Tape drive maintenance

The **tape**(C) utility performs various tape maintenance operations on all tape drives. **tape** sends commands and receives status from the tape drive. The basic form of the command is:

```
tape command [ devicefile ]
```

For example, to rewind a cartridge tape device, enter:

```
tape rewind
```

Other commands include the following:

erase	erases tape cartridge and re-tensions
reset	resets tape controller and tape drive, clears error conditions, and returns tape subsystem to power-up state
reten	re-tensions tape cartridge. Should be used periodically to remedy slack tape problems that generate an unusually large number of tape errors or, in extreme cases, actually tangle the tape in the drive.

After certain tape operations are executed, the system returns a prompt before the tape controller has finished its operation. If you enter another tape command too quickly, the message `device busy` is displayed until the tape device is finished with its previous operation.

You should clean the tape drive heads and re-tension cartridges to keep it operating error-free.

Irwin drive commands

There are certain commands that are specific to Irwin drives:

info prints information about the inserted cartridge, including whether or not it is formatted, what type of format, and write protect status

capacity or **kapacity**
displays the capacity of the inserted cartridge in 512- and 1024-byte blocks, respectively

Digital audio tape (DAT) and 8-mm tape commands

The **load** and **unload** options of the **tape**(C) command are used on DAT, 8-mm, and 9-track drives. Certain drives require a **tape load** command before any other tape commands, while most DAT drives do an automatic load when the tape cartridge is inserted.

There are certain considerations that apply specifically to using **load** and **unload** commands with DAT drives:

- The first command to a tape after power-up of the system fails. There is no error message. After a tape change, some devices issue a warning on the next tape access in the form of `Unit attention` or a similar message. In either case, a tape command has to be repeated to ensure success. However, a read or write from/to the tape succeeds if there are no other errors.
- You should wait for the drive to finish its initialization sequence before attempting to access the device. This is usually very swift on a non-DAT device but can take 30 seconds or more for a DAT.

On tape drives that do not support automatic loading and unloading of media, **tape load** and **tape unload** typically just turn the front panel light on and off.

No-unload device

DAT and 8-mm drives are configured to use a "no unload" device, which prevents the tape from being automatically unloaded (ejected) after each access. Some tape drives also require a load command after an unload. The actual unload is not obvious because the tape is not physically ejected as it is with DAT drives. (The unload is done on a **close**() after a read or write sequence.)

Tape formatting

Tape cartridges used with the mini-tape drive (ctmini) must be formatted before use. Although it is possible to format mini-tapes, it is not recommended because the results are not reliable and can cause problems when using tapes created on different drives. Preformatted mini-cartridge tapes are available and recommended for best results. See also **tape**(HW) and **tape**(C) for more information.

Tape driver error correction code (ECC) support

Tape ECC is supported on QIC-02, mini-cartridge (Irwin), and QIC-40/80 drives. ECC is not available on SCSI tape drives.

QIC-02 ECC

The QIC-02 ECC tape device node, */dev/erct0*, is automatically created when you run **mkdev tape**. To use ECC on the QIC-02, you must read and write from this device, not the normal */dev/rct0*. With tape drives that support cartridges larger than 60 Mbytes, it is a good idea to edit the */etc/default/tar* file and substitute */dev/erct0* for the normal tape device.

The error recovery scheme is 2/64, which means that two 512-byte blocks out of every 64 blocks can go bad and the driver corrects them. The probability of error with ECC is $1{:}10^{14}$. Standard drives have an error probability of $1{:}10^{9}$.

Be sure and label tapes that are created with the ECC device; these tapes cannot be read by standard devices. In addition, if transporting data from one machine to another, it is advisable to use the ECC device only if the target machine supports the ECC scheme.

Mini-cartridge and QIC-40/80

ECC encoding and decoding is automatic on the QIC-40, QIC-80, and mini-cartridge (Irwin) 80-Mbyte drives; no ECC device is necessary. For every 29K written to the tape, 3K of ECC data is written. On smaller capacity mini-cartridge drives, for every 16K written to the tape, 2K of ECC data is written.

Using floppy disks

Floppy disks are the most convenient form of storage media. Depending on your floppy disk drive, you may be able to store from 360 Kbytes to 1.4 Mbytes on a single disk. Floppy disks can be used for simple data storage in **tar, cpio,** or **dd** formats or you can make a mountable filesystem on a floppy disk. The following sections explain how to use floppies for data storage and as extra filesystem space.

Formatting floppy disks

Floppy disks must be formatted before they can be used. The command to format a floppy disk is:

format /dev/*floppy-device*

Δ **sysadmsh** users select: Media ➪ Format

The floppy device you specify in the command relates to the type of disk drive and floppy you are using. For example, if you have a high-density 5.25-inch floppy disk drive, you can use it in high-density mode (96 tracks per inch) or in low-density mode (48 tpi). If you have high-density floppies to use with your drive, the floppy device to specify is:

/dev/rfd096

In this example, *rfd* indicates the raw floppy device, *0* indicates that this is the primary floppy drive, and *96* indicates high-density mode.

NOTE You must always specify the raw device; you cannot format the block device.

Similarly, if you wish to use low-density floppies and the low-density mode of the floppy drive, the device name is:

/dev/rfd048

In this example, *48* indicates the low-density mode of floppy drive *0*.

/etc/default/format file

You can also define a default format device by adding an entry to the file */etc/default/format.* For example:

```
DEVICE=/dev/rfd096ds15
```

After adding the above line, you no longer have to specify the device name. In addition, it is possible to define that all floppies be verified, which confirms that the data on the floppy is readable. (This can also be specified on the command line with the **-v** option.) Automatic verification can be specified by the following entry:

```
VERIFY=Y
```

If this entry is placed in */etc/default/format*, all floppies formatted with the **format** command are verified. (To override verification, use the **-n** option on the command line.)

Refer to the **format**(C) manual page for more details.

Copying floppy disks

To ensure against the loss of data stored on floppy disks, any user can use the **diskcp**(C) command, or the **dd**(C) command to make copies of floppy disks on new, formatted disks.

diskcp makes use of **dd** and provides a simple interface to that program. **dd** is very powerful, and you can use it to perform many different kinds of copying.

You must copy information onto formatted disks. If you format floppies, you can use them over again without reformatting.

If you have disks that were formatted under another operating system, you must reformat them before you can use them to make copies of UNIX disks. Be aware that floppies formatted under some operating systems cannot be used under other operating systems, even with reformatting.

You can use the **format** command to format floppies. This command is described in the section "Formatting floppy disks" in this chapter. The **diskcp** command can also format floppies for you.

To copy a floppy disk using **diskcp**, do the following:

Δ **sysadmsh** users select: Media ➪ Duplicate

1. Insert the disk you want to copy, known as the *source* floppy, in drive 0, your primary floppy drive.
2. Insert another floppy in the other drive. This floppy is known as the *target* disk. Note that any information already on the target disk is destroyed.

   If you have only one disk drive, leave the source floppy in the drive. **diskcp** prompts you to remove the source disk at the correct time.

3. To format the floppy disk before the image is copied, enter the command:

   ```
   diskcp -f
   ```

   Press ⟨Return⟩.

   If your computer has dual floppy drives, enter the following command to copy the image directly on the target floppy:

   ```
   diskcp -d
   ```

   Press ⟨Return⟩.

   If you do not need to format the target floppy, simply enter:

   ```
   diskcp
   ```

   Press ⟨Return⟩.

4. Follow the instructions as they appear on your screen. Note that, with a single drive system, you are prompted to remove the source disk and insert the target disk.

To copy a disk using **dd**, follow these steps:

1. Insert the disk to be copied into floppy drive 0.
2. Insert a formatted disk into drive 1. If necessary, you can format a disk with the **format** command described under "Formatting Floppy Disks" earlier in this chapter.
3. Enter:

   dd if=/dev/fd0 of=/dev/fd1 count=*blkcount*

   Press ⟨Return⟩. The ***blkcount*** is the number of blocks on the disk to be copied. If you do not know this number, leave the **count=*blkcount*** section out of the command.

This command copies the first disk to the second, then displays a record of the number of blocks copied.

Using floppies for file storage

To use a floppy for simple file storage, first make sure that the floppy is formatted. Then, place the floppy in the floppy drive. You can use any of the standard UNIX file archiving utilities with floppy disks. These include **tar, cpio,** or **dd** formats.

tar is recommended for most file-archiving tasks. For example, to place a copy of a file on a high-density floppy disk in **tar** format, use the following command:

tar cv *filename*

Δ **sysadmsh** users select: Media ➪ Archive

For more information on **tar**, see the **tar**(C) manual page. For more information on **cpio, dd,** and **backup** formats, see the associated manual pages.

Making filesystems on floppy disks

You can make a filesystem on a floppy disk similar to how you make one on a hard disk. Filesystems on floppy disks are portable and can be mounted on any UNIX system. A special directory called */mnt* is used for mounting filesystems that do not have a specified mounting point. Note that for system security, you must be logged in as *root* to use floppy filesystems, and only *root* can mount a floppy filesystem.

To make a portable filesystem on a floppy disk, use the following procedure:

1. Log in as *root* and enter the command:

   mkdev fd

   Δ **sysadmsh** users select: Filesystems ➪ Floppy

2. You see the following menu:

```
Floppy Disk Filesystem Creation Program

        Choices for type of floppy filesystem.

        1. 48tpi, double sided, 9 sectors per track
        2. 96tpi, double sided, 15 sectors per track
        3. 135tpi, double sided, 9 sectors per track
        4. 135tpi, double sided, 18 sectors per track

Enter an option or q to quit:
```

   Enter the number of the disk type desired and press ⟨Return⟩.

3. If you have more than one floppy drive you see the following prompt:

```
Do you want to use floppy drive 0 or floppy drive 1
```

   Enter the number of the drive required and press ⟨Return⟩.

4. Next you see:

```
Choices for contents of floppy filesystem.

1. Filesystem
2. Bootable only       (96ds15 and 135ds18 only)
3. Root filesystem only (96ds15 and 135ds18 only)

Enter an option or enter q to quit:
```

   Enter **1** and press ⟨Return⟩.

5. You see the following prompt:

```
Insert a type floppy into drive 0.
Press Return to continue or enter q to quit:
```

   Press ⟨Return⟩.

6. The following prompt is displayed:

```
Would you like to format the floppy first? (y/n)
```

   If you have already formatted the floppy, enter **n** and the filesystem is immediately created. If the floppy has not yet been formatted, enter **y** and you see:

```
formatting /dev/type
track 00 head 0
```

   The track and head numbers count up as the floppy is formatted. (If */etc/default/format* contains **VERIFY=Y**, the format is also verified after formatting.)

7. Next you see a prompt for filesystem type:

```
Do you want to use the default file system type AFS (y/n)?
```

   If you respond **n**, you are asked to supply the type:

```
Please enter a file system type from <AFS, EAFS, S51K, XENIX>
or enter q to quit:
```

8. Enter the filesystem type and press ⟨Return⟩. When the process is complete, you see:

```
Filesystem creation complete.
```

9. Next you see this menu again:

```
Choices for contents of floppy filesystem.

1. Filesystem
2. Bootable only (96ds15 and 135ds18 only)
3. Root filesystem only (96ds15 and 135ds18 only)

Enter an option or enter q to quit:
```

Now enter **q** and press ⟨Return⟩ to quit. Your floppy now contains a filesystem.

Mounting a floppy filesystem

To use a floppy filesystem, you must mount it on your system. For example, a 96-tpi floppy would use the following command:

mount /dev/fd096 /mnt

Δ **sysadmsh** users select: Filesystems ➪ Mount

Note that you use the floppy device *fd096* and not *rfd096*. When you mount a floppy filesystem, you must use the name without the preceding "r". As another example, if you choose to mount a filesystem on a 48-tpi disk, use the following command:

mount /dev/fd048 /mnt

When you give the **mount** command, the shell should return a prompt. This indicates that the filesystem was successfully mounted. You can now use the **cd** command to move into the filesystem and create files there normally. When you are done and you wish to remove the floppy, give the following command:

umount /dev/fd048

Δ **sysadmsh** users select: Filesystems ➪ Unmount

Your filesystem is immediately unmounted. Your files are contained on the floppy and can be stored or transported easily.

Chapter 9

Adding mice and other graphic input devices

This chapter explains how to attach mice and other graphic input devices (such as bitpads) to your system.

Installing the hardware

Consult your hardware manufacturer's documentation for specific instructions on hardware configuration. Note the brand and type of your input device and whether it is attached to a serial port or directly to the system bus. (Keyboard mice attach to a special port found on certain systems.) For more information about the system bus, see the "Adding multiport cards, memory, and other bus cards" chapter in this guide. You need to know this information when you configure your software to accept the mouse.

Bus mice

Bus mice come with controller cards that are plugged into a slot in the computer. They have jumpers or switches that must be set to allow the computer to talk to the mouse correctly.

The manuals that come with your bus mouse should contain information about the jumpers and what the correct jumper settings should be. Although some of the jumper settings depend on the type of hardware you have, and should be set according to the manual instructions, some bus mice require specific jumper settings to work properly.

It is important to set the jumpers before you run **mkdev mouse** because resetting jumpers usually requires a software removal and reinstallation of the mouse. Be sure to read the "Adding multiport cards, memory, and other bus cards" chapter in this guide if you are unfamiliar with bus cards.

Check your system to see which interrupts are being used by other devices, so you will know the interrupts you cannot use. A list of standard interrupts is shown in Table 9.1.

Table 9-1 Typical device interrupts

Interrupt	Device
0*	Clock
1*	Console
2	Networks, tapes and others
3	Serial COM2
4	Serial COM1
5	Alternate parallel port (lp2)
6*	Floppy disk
7	Main parallel port (lp0 or lp1)

Do not use the interrupts marked with an asterisk. You should select which interrupt you want to use for your mouse and be prepared to set the jumpers to that interrupt. You can use the **hwconfig**(C) command to display your current system configuration. The interrupts in use will be under the "vec" column; be sure to avoid using an interrupt belonging to another device.

The following sections describe the correct hardware setup for specific bus mice.

Logitech bus mouse

Set the jumpers as follows:

- Set JMP1 to any interrupt that is not being used.
- Set JMP2 to jumper 1 for 30 Hz, as 60 Hz is used for DOS.

Microsoft bus mouse

Set the jumpers as follows:

- Set JMP2 and JMP3 to the proper settings listed in your manual.
- Set JMP4 to either interrupt 3, 4, or 5. Do not use the setting for interrupt 2.

Olivetti bus mouse

Set the jumper settings according to your manual.

Serial mice

Serial mice are connected to either the COM1 or COM2 port, or to a port in a multiport board. They usually require either 9-pin or 25-pin connections.

If you have a COM1 or COM2 port, plug the mouse into one of these ports. If you have a non-intelligent multiport card, plug the mouse into one of these ports.

It is important that you know the name of the port on which the mouse is to be installed. COM1 uses *tty1a,* and COM2 uses *tty2a.* Multiports that are connected to COM ports have similar names but have different letters for each port. For example, a four-port multiport board on COM1 would have device names *tty1a, tty1b, tty1c,* and *tty1d* to correspond with its four ports. Multiport cards provided with special drivers (so-called "smart cards") have their own device names; check your multiport manual to find them.

Keyboard mice

Keyboard mice simply connect to the back of your computer. They usually use 6-pin or 9-pin connectors. Check your manual to find the port to which your mouse should be connected.

Configuring a mouse

To install a mouse on your system, you must perform the following steps:

1. Install the mouse according to the manufacturer's instructions.
2. Make sure your link kit is installed and functioning correctly. The mouse drivers cannot be installed without the link kit. (The link kit is installed using **custom**(ADM).)
3. Log in as *root* and input the following command:

   mkdev mouse

   Δ **sysadmsh** users select: System ⇨ Configure ⇨ Hardware ⇨ Mouse

You see the Mouse Initialization menu:

```
        Mouse Initialization Program

        1. Display current configuration
        2. Add a mouse to the system
        3. Remove a mouse from the system
        4. Associate a terminal with an existing mouse
        5. Disassociate a terminal from an existing mouse
        6. Remove the mouse drivers from the kernel

Select an option or enter q to quit:
```

To install a mouse, select option **2** and press ⟨Return⟩. The other options allow you to change your mouse configuration at any time. For example, you can add or remove additional mice on your system or change the terminals that are allowed to receive input from an existing mouse.

4. Next, specify the type of mouse you will use. You see the menu:

```
Reading device entries...

The following mouse device types are supported:

1. Serial mouse
2. Bus mouse
3. Keyboard mouse

Select an option or enter q to return to the previous menu:
```

Enter the number corresponding to the type of mouse you wish to install and press ⟨Return⟩. If you selected 2, proceed to step 7. If you selected 3, proceed to step 9.

5. You see the following menu:

```
The following serial mouse devices are supported:

1. Logitech serial mouse
2. Logitech MouseMan: Serial or cordless version
2. Microsoft Serial Mouse
3. Mouse Systems PC II Serial Mouse
4. Mouse Systems PC Mouse

Select an option or enter q to return to the top level menu:
```

Enter the number corresponding to the mouse you wish to install and press ⟨Return⟩.

6. You see:

```
mouse_type is currently configured
to attach to the system on /dev/ttyxx

Do you want to install this mouse on a different port? (y/n)
```

   Enter **n** and press ⟨Return⟩ if you do not need to change the default port. Enter **y** followed by ⟨Return⟩ if you wish to change the default and enter a port when prompted. Proceed to step 10.

7. The bus mouse menu is displayed:

```
The following Bus mouse devices are supported:

1. Microsoft Bus Mouse
2. Olivetti Bus Mouse
3. Logitech Bus Mouse


Select an option or enter q to return to the top level menu:
```

   Enter the number corresponding to the mouse you wish to install and press ⟨Return⟩.

8. You are asked to select the configuration for the busmouse card:

```
Busmouse Configuration

1. Display current busmouse parameters
2. Modify current busmouse parameters
3. Select previous busmouse parameters
4. Select default busmouse parameters

Enter an option or q to quit:
```

   If you wish to use the default busmouse parameters, select **4**. The current parameters are displayed, and you can press **q** to quit this menu. The default busmouse selection auto-configures your busmouse. If you change the interrupt vector, then using interrupt vector 5 conflicts with a cartridge tape device (using the same vector) if both devices are in use at the same time. (This is also true of the */dev/lp2* parallel device.) Proceed to step 10.

9. You see the following menu:

```
The following Keyboard mouse devices are supported:

1. Low Resolution Keyboard Mouse
2. High Resolution Keyboard Mouse

Select an option or enter q to return to the top level:
```

Enter the number corresponding to the mouse you wish to install and press ⟨Return⟩. (See the "Changing the mouse resolution setting" and "Solving slow mouse response" sections of this chapter for further details on high resolution mice.)

10. Next, you are asked to specify the terminals and multiscreens that are allowed to accept input from the mouse. Do not specify any ttys where mice are actually connected or you will receive an error message. You may choose to allow any or all other terminals and console multiscreens to use the mouse. Simply pressing ⟨Return⟩ associates all of the console multiscreens.

    Note that only one mouse can be allowed for input on a given tty.

    For more information on sharing the mouse between several terminals or multiscreens, see "Using the mouse" later in this chapter. You see:

```
This mouse may be configured for use on any of the system's
terminals and multiscreens.  The multiscreens and terminals
that will be associated with this mouse need to be specified.

Specify them by entering, at the following prompt, all the
ttys to be associated with this mouse.  Entering the word
"multiscreen" will associate all of the console multiscreens.

Enter a list of terminals (e.g. tty1a tty2a multiscreen)
or enter q to quit.  The default is multiscreen

Press return when finished:
```

    If only the multiscreens are to be associated with the mouse, just press ⟨Return⟩. If you need to associate other devices, enter them and press ⟨Return⟩. Now you see:

```
Do you want to use the mouse_type on
any other terminals? (y/n)
```

    Note that in this example ***mouse_type*** is replaced with the brand or type of mouse you specified earlier in the procedure. Respond **n** if no other terminals are allowed to receive mouse input. If you answer **y**, you are returned to the screen prompting for a list of terminals.

11. You see the following messages, which may take a few minutes to appear on your screen:

```
You must create a new kernel to effect the driver change you specified.
Do you wish to create a new kernel now? (y/n)
```

    Answer **y** to add the mouse device driver to your kernel.

12. Next, you see:

```
     The UNIX operating system will now be rebuilt.
     This will take a few minutes.  Please wait.

     Root for this system build is /.
```

    As part of the linking process, you see the following messages:

```
     The UNIX kernel has been rebuilt.
Do you want this kernel to boot by default? (y/n)
```

    Answer **y** if you want this kernel to be used every time you boot the system.

13. The following is displayed:

```
Backing up /unix to /unix.old.
Installing new /unix.

The kernel environment includes device node files and /etc/inittab.
The new kernel may require changes to /etc/inittab or device nodes.

Do you want the kernel environment rebuilt? (y/n)
```

    Enter **y**.

14. The following is displayed:

```
The new kernel has been successfully linked and installed.
To activate it, reboot your system.

Setting up new kernel environment.
```

    You have now installed the mouse drivers in your kernel.

15. Finally, you are returned to the main mouse menu again. If you have nc changes to make to your mouse configuration at this time, enter **q** to quit and press ⟨Return⟩.

16. Use the **shutdown**(ADM) command (or **haltsys**(ADM) if you are in single-user mode) to shut down the system and reboot.

You can invoke **mkdev mouse** at any time to allow or prevent input on different terminals, remove mice, or check your current configuration.

NOTE Many system utilities and applications (for example, Office Portfolio) which use a mouse (for example, **usemouse**(C)) require pseudo ttys to be installed on the system. If you wish to use such an application or use **usemouse**(C) to test your mouse as described below, please refer to the section on using the **mkdev ptty** command in the "Administering serial terminals" chapter of this guide.

Testing a mouse

Use the following procedure to test your mouse.

NOTE Pseudo-ttys must be installed on the system in order to use **usemouse**(C).

1. Log in as *root* in multiuser mode.
2. Enter the following at the system prompt:

   usemouse -t vi -c "view /etc/termcap"
3. Observe the cursor as you move the mouse. The cursor should move as the mouse moves.

   Pressing each mouse button should result in the following actions:
   - Left button moves the cursor to the beginning of the file.
   - Middle button deletes the current character.
   - Right button moves the cursor to the last line of the file.
4. To stop the **usemouse** utility, enter the following:

   :q!

If invoking **usemouse** does not produce the cursor behavior described here, or you see the following error messages:

```
Open event driver failed:: No such file or directory
Open event driver failed:: Not enough space...giving up
```

Your mouse is not installed incorrectly. Check the following:

- First, verify that your mouse is one of the supported mice listed in your *Release Notes.*
- Make certain the hardware is seated properly and the cable is attached securely.
- If you are using a busmouse, verify that the busmouse card is recognized during boot up of your system and that there is no conflict with the interrupt vector or base address. Check the hardware configuration information using the **hwconfig**(C) command.
- If you are using a serial mouse, verify that the serial card to which your mouse is attached is recognized during boot up of your system (by running **hwconfig**). If the mouse is on a multiport board that uses its own drivers, make certain the board works. Try the mouse on a COM port to eliminate the possibility of a third-party driver being the problem. You can also try connecting a terminal and testing it as described in the "Administering serial terminals" chapter of this guide.

Removing a mouse

Removal of any mouse or the mouse drivers on your system is an exact reversal of the process of installing a mouse. Choose the menu options to remove rather than to add a mouse.

Using the mouse

Use of a mouse is automatic. If a program or utility accepts mouse input and the terminal is allowed to use the mouse, you simply invoke the program and the mouse works. If the terminal or multiscreen is not allowed to use the mouse, or the program is not configured to accept mouse input, using the mouse has no effect.

Using the mouse with multiscreens

Multiscreens (on monitors attached to video cards in the bus) provide the most convenient method for using the mouse. If a mouse is associated with the multiscreens on your main system console (typically, a monitor attached to a video card in the system bus), the mouse input is associated with the current active multiscreen. For example, if your system has four multiscreens enabled on the main system console and all those screens are allowed to use the mouse, the input from the mouse goes to the program running on the active multiscreen.

Remember that programs that do not accept mouse input are unaffected by moving the mouse, even on a mouse-allowed multiscreen.

Serial (terminal) multiscreens and serial consoles can also be configured to use the mouse.

Using the mouse on serial terminals

When you install the mouse, you are prompted to list the ttys that are allowed to use mouse input. You can allow terminals on serial lines to use the mouse just as you allow multiscreens. You must not specify any ttys where mice are physically connected.

Sharing a mouse with several terminals

When the mouse is shared among several terminals, the mouse is associated with a tty on a "first-come, first-served" basis. The first user to invoke a mouse-enabled program has the mouse for the duration of that program. For another user to use the mouse, the first user must quit the program. (This closes the input queue from the mouse.) Then, the next user for the mouse can invoke the program and open the line for input from the mouse.

Note that other users on ttys allowed to use a mouse can use programs that accept mouse input while the mouse is busy. If the mouse is busy, the programs are unable to receive input from the mouse but should otherwise function normally.

Using a mouse with keyboard-based programs

The **usemouse**(C) utility maps mouse movements and operations to key strokes used by keyboard-based programs. Refer to the **usemouse**(C) manual page for complete information.

WARNING Do not use the **usemouse** utility while in single-user (maintenance) mode.

Configuring a bitpad

To install a bitpad on your system, you must perform the following steps:

1. Install the bitpad according to the manufacturer's instructions.
2. Make sure your link kit is installed and functioning correctly. The bitpad drivers cannot be installed without the link kit. (The link kit is installed using **custom**(ADM).)
3. Log in as *root* and input the following command:

   mkdev bitpad

4. You see the Bitpad Initialization menu:

```
           Bitpad Initialization Program

           1. Display current configuration
           2. Add a bitpad to the system
           3. Remove a bitpad from the system
           4. Associate a terminal with an existing bitpad
           5. Disassociate a terminal from an existing bitpad
           6. Remove the bitpad drivers from the kernel

 Select an option or enter q to quit:
```

   To install a bitpad, select option **2** and press ⟨Return⟩. The other options allow you to change your bitpad configuration at any time. For example, you can add or remove additional bitpads on your system or change the terminals that are allowed to receive input from an existing bitpad.

5. Next, specify the type of bitpad you will use. You see the menu:

```
Reading device entries...

The following bitpad device types are supported:

1. Serial bitpad

Select an option or enter q to return to the previous menu:
```

   Enter **1** and press ⟨Return⟩.

6. You see the following menu:

```
The following serial bitpad devices are supported:

list of bitpad devices

Select an option or enter q to return to the top level menu:
```

   Enter the number corresponding to the bitpad you wish to install and press ⟨Return⟩.

7. You see:

```
bitpad_type is currently configured
to attach to the system on /dev/tty

Do you want to install this bitpad on a different port? (y/n)
```

   Enter **n** and press ⟨Return⟩ if you do not need to change the default port. Enter **y** followed by ⟨Return⟩ if you wish to change the default and enter a port when prompted.

8. Next, you are asked to specify the terminals and multiscreens that are allowed to accept input from the bitpad. Do not attempt to allow bitpad input on any tty where any bitpads are physically connected or you receive an error message. You may choose to allow any or all other terminals and console multiscreens to use the bitpad. Simply pressing ⟨Return⟩ associates all of the console multiscreens.

   Note that only one bitpad can be allowed for input on a given tty.

Mice and bitpads can be shared. For more information on sharing these devices between several terminals or multiscreens, see "Using the mouse" earlier in this chapter. You see:

```
This bitpad may be configured for use on any of the system's
terminals and multiscreens.  The multiscreens and terminals
that will be associated with this bitpad need to be specified.

Specify them by entering, at the following prompt, all the
ttys to be associated with this bitpad.  Entering the word
"multiscreen" will associate all of the console multiscreens.

Enter a list of terminals (e.g. tty1a tty2a multiscreen)
or enter q to quit.  The default is multiscreen

Press return when finished:
```

If only the multiscreens are to be associated with the bitpad, just press ⟨Return⟩. If you need to associate other devices, enter them and press ⟨Return⟩. Now you see:

```
Do you want to use the <bitpad_type> on any other terminals?
(y/n)
```

Note that in this example ***bitpad_type*** is replaced with the brand or type of bitpad you specified earlier in the procedure. Respond **n** if no other terminals are allowed to receive bitpad input. If you answer **y**, you are returned to the screen prompting for a list of terminals.

9. You see the following messages, which may take a few minutes to appear on your screen:

```
You must create a new kernel to effect the driver change you specified.
Do you wish to create a new kernel now? (y/n)
```

Answer **y** to add the bitpad device driver to your kernel.

10. Next, you see:

```
     The UNIX operating system will now be rebuilt.
     This will take a few minutes.  Please wait.

     Root for this system build is /.
```

As part of the linking process, you see the following messages:

```
     The UNIX kernel has been rebuilt.

Do you want this kernel to boot by default? (y/n)
```

Answer **y** if you want this kernel to be used every time you boot the system.

11. The following is displayed:

```
Backing up /unix to /unix.old.
Installing new /unix.

The kernel environment includes device node files and /etc/inittab.
The new kernel may require changes to /etc/inittab or device nodes.

Do you want the kernel environment rebuilt? (y/n)
```

    Enter **y**.

12. The following is displayed:

```
The new kernel has been successfully linked and installed.
To activate it, reboot your system.

Setting up new kernel environment.
```

    You have now installed the bitpad drivers in your kernel.

13. Finally, you are returned to the main bitpad menu again. If you have no changes to make to your bitpad configuration at this time, enter **q** to quit and press ⟨Return⟩.
14. Use the **shutdown**(ADM) command (or **haltsys**(ADM) if you are in single-user mode) to shut down the system and reboot.

You can invoke **mkdev bitpad** at any time to allow or prevent input on different terminals, remove a bitpad, or check your current configuration.

Changing the mouse resolution setting

The file */etc/conf/pack.d/kbmouse/space.c* contains two parameters that affect the performance of the high resolution mouse, **kbm_resolution** and **kbm_poll**. You can use a text editor to edit the */etc/conf/pack.d/kbmouse/space.c* file and change the **kbm_resolution** parameter to adjust the resolution setting. The **kbm_poll** parameter, however, should only be changed automatically by the **mkdev mouse** utility under normal circumstances. See the section "Solving slow mouse response" for more information about **kbm_poll**.

The **kbm_resolution** parameter determines how many reports, or counts, are made from the mouse to the mouse driver each time that you move the mouse one millimeter. Increasing the number of counts per millimeter increases the sensitivity of mouse performance. The allowable values for **kbm_resolution** are:

Table 9-2 High-Resolution Keyboard Mouse

Counts/millimeter	Parameter
1	kbm_resolution=0
3	kbm_resolution=1
6	kbm_resolution=2
12	kbm_resolution=3

The **kbm_resolution** parameter is set to 1 by default (for 3 counts/millimeter) in *etc/conf/pack.d/kbmouse/space.c*.

The parameters for a keyboard mouse do not correspond to the same counts/millimeter parameters as for a high-resolution keyboard mouse:

Table 9-3 Keyboard Mouse

Counts/millimeter	Parameter
1	kbm_resolution=0
2	kbm_resolution=1
4	kbm_resolution=2
8	kbm_resolution=3

The default **kbm_resolution** value for a keyboard mouse is 3, specifying 8 counts/millimeter. The same value specifies 12 counts/millimeter for a high-resolution keyboard mouse, which causes the high-resolution keyboard mouse to behave too sensitively. Therefore, you must use the **mkdev mouse** utility when you add or remove a high-resolution mouse. **mkdev mouse** changes the default parameters in *etc/conf/pack.d/kbmouse/space.c* to the appropriate values for the mouse that you are configuring. You must answer **y** when **mkdev mouse** prompts you if you want to relink the kernel if you want to effect the new parameter values.

Solving slow mouse response

If your mouse responds too slowly you should use a text editor to edit the file */etc/conf/pack.d/kbmouse/space.c*. Lower the **kbm_poll** value to improve mouse response. The minimum value is 0x5. For example, if the value 0xb0 produces very slow mouse response, try 0xa0, 0x90, 0x80, and so on in turn until you have satisfactory mouse performance.

NOTE Some **kbm_poll** values lower than 0xb0 may cause your system to freeze. If this occurs, raise the **kbm_poll** value.

After modifying parameters in */etc/conf/pack.d/kbmouse/space.c*, you must relink the kernel. To do this, execute **./link_unix** from the */etc/conf/cf.d* directory and reboot the system.

Chapter 10
Adding hard disks and CD-ROM drives

This chapter explains how to add and maintain conventional hard disks and CD-ROM drives. CD-ROM drives are essentially read-only filesystems that can be accessed just like a hard disk.

Adding secondary hard disks

When your system suffers from chronic lack of space, you probably need to add a hard disk to give the system extra space for storing user files and directories. The following types of secondary hard disks and controllers are supported:

- Standard disk support (ST506, ESDI, MFM, RLL, IDE)
- SCSI host adapters
- Compaq IDA/Intelligent Array Expansion Controllers

Three configurations are possible:

- root disk on a SCSI host adapter with an option to add one SCSI host adapter, each supporting up to seven controllers (Target IDs), and each SCSI controller supporting up to eight devices
- root disk on an ST506/ESDI controller with an option to add one ST506/ESDI controller, each supporting two ST506 or ESDI disks and up to two SCSI host adapters (which can be configured as in the first option)
- root disk on an IDA controller with an option to add a maximum of four logical pairs on two IDA controllers.

Figure 10-1 illustrates a configuration of the second type.

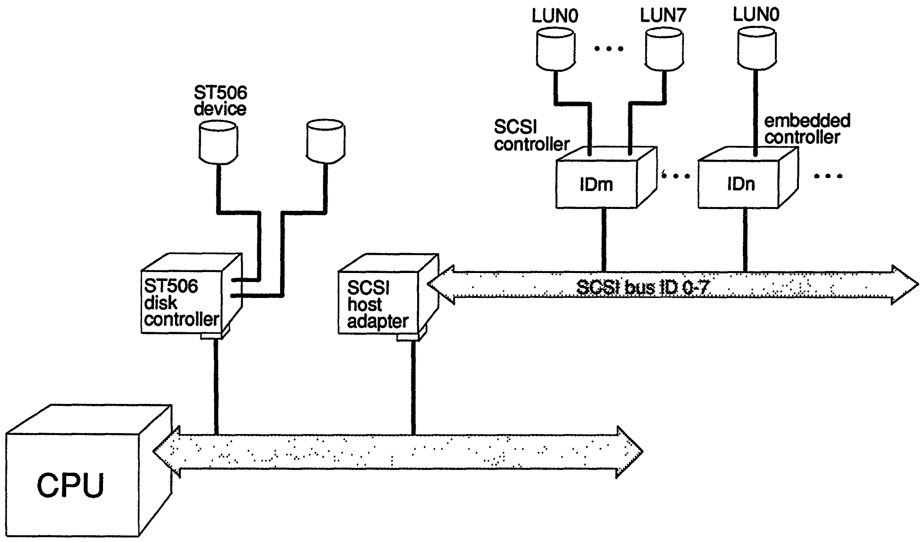


Figure 10-1 ST506 and SCSI configuration examples

A SCSI host adapter (HA) translates signals from the CPU bus to the SCSI bus. A SCSI controller is known as a "Target ID." A SCSI device is referenced by a logical unit number (LUN).

When you installed the operating system and initialized your root disk, the root disk was configured as the first hard disk on the first controller (for ST506 or ESDI disks) or first host adapter (for SCSI disks).

Although the basic procedure for adding a disk is common to all types of disks, you occasionally need to perform somewhat different steps based on the type of disk you are installing. Throughout the procedure, the differing steps are clearly indicated.

Before you start

This section explains the syntax used for the **mkdev hd** command used to configure and add hard disks. You should use this section to determine what command-line options are necessary to configure your ST506, ESDI, or IDA logical disks. When you have chosen the proper syntax, you can proceed to "Installing the hard disk," later in this chapter. In the case of SCSI disks, you must read this section, invoke the **mkdev hd** command as instructed, then proceed to "Installing the hard disk" where you invoke the same command a *second* time. This is necessary because the SCSI configuration files must be prepared in the first pass and the disk initialized in the second.

NOTE Your hardware must first be jumpered or configured according to the documentation provided with your machine. Many machines include a setup or configuration disk that must be booted and used to configure your system; this must be done before running **mkdev hd**.

Configuring a hard disk

You need to decide how you want to configure the disk so you can provide that information to the installation utility.

Standard and IDA disk controllers

To configure a standard (ST506 or ESDI) disk and controller or one attached to Compaq IDA/Intelligent Array Expansion controller with the **mkdev hd** command, you must know which disk controller supports the new disk and whether it is the first or second disk on the controller. There is a limit of two disks per controller. (IDA controllers have a limit of two "logical" disks per controller, which can consist of more than one disk each.) The command syntax is:

mkdev hd *disk controller*

Numbering of disks and controllers starts at 0. See Table 10.1.

Table 10-1 Standard and IDA controller commands

Command	Disk being added
mkdev hd 0 0	first disk on first controller (install time)
mkdev hd 1 0	second disk on first controller
mkdev hd 0 1	first disk on second controller
mkdev hd 1 1	second disk on second controller

The controller number can also be referenced using a more literal syntax; for example, ST506-0 refers to the first (root) ST506 controller. Thus, a command line using this syntax looks like this:

mkdev hd 0 ST506-0

The other available codes are ESDI and IDA.

SCSI host adapters

To configure a SCSI device with the **mkdev hd** command, you must know:

- *ID Number*. This is the number of the controller on the host adapter (0-7); the host adapter itself is usually ID 7, giving it the highest priority on the SCSI bus.
- *Host Adapter Type*. This is the code that refers to the type of host adapter. Several types are supported. For example: **ad** refers to an Adaptec host adapter (**ad** is also the default type).

- ***Host Adapter Number.*** This shows which host adapter the device is connected to.
- ***Logical Unit Number.*** This is the number of the device (0-7) on the "Target ID"; on an embedded controller (where the controller and the device are one physical unit), the LUN is usually 0.

Refer to Figure 10-1 for a pictorial representation of each value. You can choose to be prompted for these values, or you can provide them on the command line. Use the following syntax to specify the configuration:

mkdev hd ***id ha lun hatype***

The arguments are as follows:

id	number from 0-7
ha	host adapter number
lun	number from 0-7
hatype	type of host adapter

The instructions that follow assume that the **mkdev hd** command is invoked without arguments. If you do provide all the information on the command line, you can skip to the last step of the procedure that follows. You must provide the identical arguments when you invoke **mkdev hd** the second time.

To add your SCSI disk, follow these steps:

1. Enter the following command:

   mkdev hd ***arguments***

   Δ **sysadmsh** users select: System ⇨ Hardware ⇨ HardDisk
2. If your root disk is attached to an ST506 controller, you see the following:

```
Your root hard disk is attached to an ST506 controller.

Pick one of the choices below or you may quit and
invoke mkdev hd -u for a detailed usage message.

                     1) Add a hard disk to ST506 controller
                     2) Add a hard disk to SCSI controller
                     3) Add a hard disk to an IDA controller (EISA)

Enter 1, 2, 3 or enter q to quit:
```

   Enter **2** and press ⟨Return⟩.

3. If your root disk is attached to a SCSI controller, you see the following:

```
Your root disk is attached to a SCSI controller.

The only available choice is to add another SCSI disk.
Do you want to add another SCSI disk?
```

   Enter y and press ⟨Return⟩.

4. Now you must identify the type of the adapter that the disk is attached to:

```
Enter the prefix of the SCSI host adapter that supports this device
or press ⟨Return⟩ for the default: type
Enter h for a list of host adapters or enter q to quit:
```

   Enter the code that describes your host adapter: for example, **ad** for Adaptec. Entering **h** displays a list of supported adapters.

5. Now you are prompted for the number of the adapter:

```
Which type SCSI host adapter supports this device?
Select 0-n or q to quit:
```

   Enter **0** if it is the first ***type*** host adapter or **1** for the second ***type*** host adapter.

6. If you are adding a device to a previously configured host adapter, skip to step 9. If you are configuring a secondary host adapter of a type already installed, you see the following:

```
hatype SCSI host adapter n is not configured.
Would you like to update the link-kit?
```

   The kernel already recognizes one of each type of host adapter; an additional entry must be created for the new one. Respond **y** and press ⟨Return⟩.

7. You may be prompted for one or more of the following, depending on your host adapter type (the system attempts to autoconfigure any values it can obtain from default files and other adapters installed of the same type):

```
What is the interrupt vector for this adapter?

What is the start IO address (hex) for this adapter?

What is the end IO address (hex) for this adapter?

What is the start controller memory address (hex) for this adapter?

What is the end controller memory address (hex) for this adapter?
```

   The hardware appendix of the *Release Notes* contains a table of default SCSI host adapter configuration parameters. If you do not have a given type of adapter installed, you can probably use the values for another adapter. (You should check for conflicts before entering values; other hardware devices may already be using the same interrupt vector, I/O address, and so forth.) Enter each value as prompted and press ⟨Return⟩.

8. You are asked to confirm the values you supplied. Then the following prompt is displayed:

```
The following parameters will be used to configure
hatype SCSI host adapter ha

list of parameters

Update the link-kit?
```

   If you need to make corrections, enter **n** and press ⟨Return⟩, and then press ⟨Del⟩ to exit **mkdev hd** and start over. If the values are correct, enter **y** and press ⟨Return⟩ to add these values to the link kit.

9. Next you see:

```
What is the Target ID for this device?
Select 0-7, or 'h' for help, or 'q' to quit:
```

   Enter the number of the controller attached to the adapter.

10. You are then prompted:

```
What is the LUN of this device?  Press <RETURN to use the default: 0
Select 0-7, or 'h' for help, or 'q' to quit:
```

Enter the number of the device attached to the controller. With most disks, the controller and the device are a single unit, in which case the Logical Unit Number is 0. Non-embedded controllers can support up to eight SCSI devices. In this case, LUN numbers are determined by jumper settings on each device. The valid range of LUN numbers is 0-7.

11. The information you supplied earlier for the device configuration is then displayed as in the following example. You are asked to confirm that you wish to update the configuration:

```
You are about to add the following SCSI device:

Host
Adapter          Adapter
Type    Device  Number  ID      LUN
-----------------------------------
ad      Sdsk    0       1       0

Update SCSI configuration (y/n)?
```

If the information is correct, enter **y** and press ⟨Return⟩.

12. Now that the data is entered, the program acknowledges the information and prompts for relinking of the kernel:

```
The SCSI configuration file has been updated.

A new kernel must be built and rebooted before disk configuration can continue.
Would you like to relink at this time? (y/n)
```

The kernel must be reconfigured to recognize the new disk. You are given the option of not relinking in case you are adding a number of devices. This way the kernel needs to be relinked only once before running **mkdev hd** the second time. After the reconfiguration is complete, the following message is displayed:

```
After the system is rebooted with the new kernel,
reinvoke mkdev hd to initialize the new SCSI hard disk.
```

You now have configured the necessary software support for your new disk. You can now proceed to the next section, "Preparing the hardware."

WARNING You must relink the kernel and reboot before running **mkdev hd** for the second time.

Preparing the hardware

Hard disks that do not have matching entries in the ROM tables are supported through software. When adding secondary hardware, you must change some of the switch settings on the host adapter, Target ID, and disk. "Disk controllers and host adapters" in the *Release Notes* explains what these settings should be. Check the hardware manual for your hard disk drive and the computer for instructions.

When you change the settings on a SCSI device with an embedded controller, remember to use the SCSI ID number, not the LUN. The LUN on an embedded controller is 0, because it is the first and only device on the controller.

Before adding the new disk, you must know how to connect it to the computer. Connecting the hard disk is explained in the hardware manual provided with the disk.

Make sure the additional drive is formatted and passes the manufacturer diagnostics before installing the system. If it does not pass the diagnostic tests, you cannot use it with your system.

Installing the hard disk

These are the steps to install another hard disk with one UNIX filesystem and no DOS area:

1. After you have connected the hard disk and booted the system, enter system maintenance mode and use the appropriate form of the **mkdev** command, specifying the required configuration information on the command line:

   mkdev hd ***arguments***

   Δ **sysadmsh** users select: System ➪ Hardware ➪ HardDisk

2. If your root disk is attached to an ST506 or IDA controller, you see the following, where ***type*** is replaced by ST506 or IDA:

```
Your root hard disk is attached to an type controller.

Pick one of the choices below or you may quit and
invoke mkdev hd -u for a detailed usage message.

                   1) Add a hard disk to ST506 controller
                   2) Add a hard disk to SCSI controller
                   3) Add a hard disk to an IDA controller (EISA)

Enter 1, 2, 3 or enter q to quit:
```

   Enter a number and press ⟨Return⟩. If you are adding an ST506 disk, proceed to step 8. If you are adding a SCSI disk, proceed to step 6.

3. You see:

```
Note: This kernel is configured to support n hard disks.
If your system will have more than n disks, the kernel
must be reconfigured.
```

   If you plan to install more than the number of disks indicated, you must first increase the value of the **NDISK** kernel parameter. See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on increasing this parameter. You can then reboot the system and invoke **mkdev hd** again.

4. Next, you see:

```
Will this be the first or second logical disk on this controller?
Enter 1 (first) or 2 (second):
```

   Enter the number of the logical disk and press ⟨Return⟩.

5. You are then asked to indicate the controller number:

```
Which controller will this logical disk attach to?
Enter 1 (first controller), 2 (second controller), etc.
```

   Enter the IDA controller number and press ⟨Return⟩. Proceed to step 10.

6. If your root disk is attached to a SCSI controller, you see the following:

```
Your root disk is attached to a SCSI controller.

The only available choice is to add another SCSI disk.
Do you want to add another SCSI disk?
(For detailed usage message, pick 'n' to
exit this script and invoke mkdev hd -u) (y/n)
```

   Enter **y** and press ⟨Return⟩.

7. You are then prompted to enter the same controller, host adapter, and LUN information you provided earlier:

```
Enter the prefix of the SCSI host adapter that supports this device
or press <Return> for the default: type
Enter h for a list of host adapters

Which type SCSI host adapter supports this device?
Select 0-n, or 'h' for help, or 'q' to quit:

What is the Target ID for this device?
Select 0-7, or 'h' for help, or 'q' to quit:

What is the LUN of this device?
Select 0-7, or 'h' for help, or 'q' to quit:
```

   Enter each as instructed and proceed to step 10.

8. If you are adding an ST506 or ESDI disk, you see the following:

```
Will this disk be the first or second disk on this controller?"
Enter 1 (first) or 2 (second):
```

   Enter a number and press ⟨Return⟩.

9. If you are adding an ST506 disk only, you see the following:

```
Will this disk attach to the first or second ST506 controller?
Enter 1 (first) or 2 (second):
```

   Enter a number and press ⟨Return⟩.

10. The following prompt is displayed:

```
During installation you may choose to overwrite all
or part of the present contents of your hard disk.

Do you wish to continue? (y/n)
```

   Enter y and press ⟨Return⟩.

11. If you have a SCSI controller, you see the following message:

```
The hard disk installation program will now invoke /etc/fdisk.
Entering 'q' at the following menu will exit /etc/fdisk.
and the hard disk installation will continue.

If you wish to exit the entire installation at this menu,
press the <Del> key.
```

   Skip to step 15.

   NOTE The SCSI installation skips steps 12-14.

12. If you have an ST506/ESDI (standard interface) controller, or an IDA controller, you see the following message and prompt:

```
The hard disk installation will now invoke /etc/dkinit.
Entering 'q' at the following menu will exit /etc/dkinit,
and the hard disk installation will continue.

If you wish to exit the entire installation at this menu,
press the ⟨Del⟩ key.

Hard Disk Drive 1 Configuration

         1. Display current disk parameters
         2. Modify current disk parameters
         3. Select default disk parameters

Enter an option or 'q' to quit:
```

The **dkinit** menu is intended for unusual or nonstandard disks. If you have a standard hard disk, one that is supported by your computer hardware or special motherboard ROM, enter **3** followed by ⟨Return⟩ and then **q** and ⟨Return⟩ to continue the installation.

Entering **q** at this point selects the default parameters for your hard disk. Unless you know that your disk is nonstandard, assume that it is standard and enter **q** and press ⟨Return⟩ to continue your installation. Skip to step 15.

NOTE If you are not sure if your disk is nonstandard, check the default parameters using option **1** of the **dkinit** menu. Calculate the size of your disk in bytes using the following calculation:

size = ***cylinders*** x ***heads*** x ***sectors/track*** x 512

In addition, some drives are sold by formatted size, others by unformatted size. The formatted size of a drive is approximately 85% of its unformatted size. Note that the parameters displayed by **dkinit** may not match the drive manufacturer's documentation. Some controllers have optional "translation", "mapping", or "63-sector" modes. If one of those modes was chosen during low-level formatting, the UNIX system must be initialized with the translated parameters and not those of the physical drive. In all cases, the known size of the drive should approximately match the size calculated above from the disk parameters.

If your disk is nonstandard, you must enter information that overrides the ROM disk configuration information, replacing it with the new information. If you are unsure of what parameters to enter for your nonstandard disk, contact your disk manufacturer for this information.

If you enter **1** or **2**, you see the following display:

```
Disk Parameters    Values
1. Cylinders       value
2. Heads           value
3. Write Reduce    value
4. Write Precomp   value
5. Ecc             value
6. Control         value
7. Landing Zone    value
8. Sectors/track   value
```

In the actual display, ***value*** is replaced with the default value for that variable.

NOTE The "Cylinders" value refers to the number of cylinders on the entire hard disk and should not be confused with the number of cylinders you allocated (or intend to allocate) to a given partition.

If you entered a **1**, you now see the first menu again. If you entered a **2**, you are now prompted:

```
Enter a parameter to modify or q to return to the main menu:
```

13. Enter any number from 1 to 8 to change the disk parameters, or **q** to return to the previous menu. You see the following:

```
Enter the new value or ⟨Return⟩ to use the existing value:
```

If you wish to change the value, enter a new value now or press ⟨Return⟩ to use the existing value.

14. After you finish changing the disk parameters, enter **q** to return to the main menu. Next, enter **q** again to save the changes you made. Exiting from **dkinit** by entering **q** overwrites any parameters you changed with the new values. If you wish to restore the default parameters after making modifications, enter **3** from the first menu.
15. The installation program next runs the **fdisk**(ADM) utility to partition the hard disk. You can also partition your disk to support DOS on the same hard disk (if you have DOS already installed), or you can use the whole disk for your UNIX system.

After a moment, the **fdisk** menu appears on the screen. You see this option list:

```
      1. Display Partition Table
      2. Use Entire Disk for UNIX
      3. Use Rest of Disk for UNIX
      4. Create UNIX Partition
      5. Activate Partition
      6. Delete Partition

Enter your choice or q to quit:
```

Select option **1** and press ⟨Return⟩.

If you have never installed an operating system on your disk, you see a table similar to this:

```
Current Hard Disk Drive: /dev/rhd10
```

Partition	Status	Type	Start	End	Size

```
Total disk size: 1220 tracks (5 reserved for masterboot and diagnostics)
Press ⟨Return⟩ to continue
```

If you have previously installed an operating system on your disk, the **fdisk** table is filled in. DOS is usually displayed as partition number 4.

16. Press ⟨Return⟩ to return to the main **fdisk** menu. If you would like the UNIX partition to occupy the whole disk, select option **2**. After you have made your selection, quit out of the **fdisk** menu by entering **q**. If any other operating systems were previously installed on your system, you also see the following warning message:

```
Warning! All data on your disk will be lost!
Do you wish to continue? (y/n)
```

Enter **y** and press ⟨Return⟩ only if you want your UNIX system to occupy the whole disk. This ensures that **fdisk** partitions the whole disk.

NOTE If you choose option 3, which allocates the remainder of the hard disk for the UNIX system, you must next activate the UNIX partition by selecting option 5. If you do not activate the UNIX partition, your first partition is activated.

Most computers have diagnostic programs that write to the last cylinder of the hard disk. This means that the last cylinder should not be allocated to a partition. The last cylinder is not allocated when you choose option 2 from the **fdisk** menu. If you choose option 4, you should not allocate the last cylinder of the hard disk to the UNIX partition.

17. Press ⟨Return⟩, and you see the main **fdisk** menu. You have now set up the partition(s) on your hard disk. To continue with the next step in the installation procedure, enter **q** and press ⟨Return⟩.

    If you have an ST506/ESDI controller, continue with step 18.

    If you have an IDA controller or SCSI host adapter, skip to step 26.

    NOTE Bad tracks are handled automatically by IDA controllers and SCSI host adapters, and steps 18-25 are omitted.

18. Now you see a menu from the program **badtrk**(ADM). With the **badtrk** program, you can scan your hard disk for defective tracks. The program maps any flawed locations to good tracks elsewhere on the disk. It also creates a bad track table, which is a list of all the bad tracks on your hard disk.

    The main **badtrk** menu looks like this:

```
1. Print Current Bad Track Table
2. Scan Disk (You may choose Read-Only or Destructive later)
3. Add Entries to Current Bad Track Table by Cylinder/Head Number
4. Add Entries to Current Bad Track Table by Sector Number
5. Delete Entries Individually from Current Bad Track Table
6. Delete All Entries from Bad Track Table

Enter your choice or q to quit:
```

    Enter **2**, then press ⟨Return⟩.

19. You see the following submenu:

```
1. Scan entire UNIX partition
2. Scan a specified range of tracks
3. Scan a specified filesystem

Enter your choice or q to quit:
```

    Select option **1**.

20. After you select the area you want scanned, you are given the following choices:

```
1. Quick scan (approximately 7 megabytes/min)
2. Thorough scan (approximately 1 megabyte/min)

Enter your choice or q to quit:
```

    Select option **2**.

21. You are prompted:

```
Do you want this to be a destructive scan? (y/n)
```

Enter y. You are warned:

```
This will destroy the present contents of the region you are scanning.
Do you wish to continue? (y/n)
```

Enter **y** and press ⟨Return⟩. You see the following message:

```
Scanning in progress, press 'q' to interrupt at any time.
```

22. After you respond to the above prompts, the program scans the active partition of the new disk for flaws. The larger your disk, the longer the scanning process takes, so a very large disk may take a while.

    As **badtrk** scans the disk, it displays the number of each track it examines, and the percentage of the disk already scanned. Pressing the **q** key at any time interrupts the scan. If you press **q** to interrupt the scan, you do not need to press ⟨Return⟩. You are then prompted to continue scanning or to return to the main menu.

    Whenever **badtrk** finds a defective track, it lists the location of that track using both the sector number and cylinder or head conventions. Defective track information is entered into the table and displayed on the screen. Here is an example of a bad track:

```
WARNING : wd: on fixed disk ctlr=0 dev=0/47 block=31434 cmd=00000020
    status=00005180, sector = 62899, cylinder/head = 483/4
```

23. When the scan is complete, the menu reappears. Select option **1** to see the results of the scan. Your bad track table looks something like this:

```
Defective Tracks

         Cylinder   Head   Sector Number(s)
 1.        190        3       12971-12987

Press ⟨Return⟩ to continue
```

Press ⟨Return⟩ to return to the main menu.

NOTE If there is a flaw in the first few tracks of the UNIX partition, you are returned to the **fdisk** utility (see the previous installation step). Repartition the disk with **fdisk** so that the UNIX partition no longer includes the defective tracks. You have to experiment to determine how many tracks to exclude. Leave these defective tracks unassigned to any operating system. When you leave **fdisk**, **badtrk** is run again and you should scan the disk for further flaws.

This process continues until **badtrk** finds no flaws in the first few tracks.

24. To exit **badtrk**, enter **q** and press ⟨Return⟩.
25. You are next prompted for the number of tracks to allocate as replacements for those tracks that are flawed. You should allocate at least as many as the recommended number. Enter the number or just press ⟨Return⟩ to use the recommended number that is displayed:

```
Enter the number of bad tracks to allocate space for
(or press ⟨Return⟩ to use the recommended value of n):
```

If you press ⟨Return⟩ and do not enter an alternate value, **badtrk** allocates the recommended number of tracks as replacements. This number is based on the number of bad tracks currently in the table, plus an allowance for tracks that may go bad in the future. If you ever exceed the number of allocated bad tracks, you must reinstall the system. Next, you see a prompt from **divvy**(ADM). The **divvy** program divides a partition into filesystems. You can create up to seven divisions on a single partition, and name them anything you like.

NOTE The maximum filesystem size is 2 gigabytes. When **divvy** is run on a logical drive that is greater than 2 gigabytes, it automatically calculates the minimum number of **divvy** partitions that can exist on this logical drive. For example, on an IDA controller that has a logical drive configured as 2.6 gigabytes, there is a minimum of two 1.3-gigabyte filesystems. You can modify the **divvy** partition so that one filesystem is 2 gigabytes and one is 600 megabytes. Remember that a block is 1024 bytes. To convert blocks to megabytes, simply divide by 1024.

26. You see the main **divvy** menu and a display that shows how your disk is divided similar to the one here:

```
+----------+-------------+--------+---+-------------+------------+
| Name     | Type        | New FS | # | First Block | Last Block |
+----------+-------------+--------+---+-------------+------------+
|          | NOT USED    |  no    | 0 |            0|       39011|
|          | NOT USED    |  no    | 1 |        39012|       41511|
|          | NOT USED    |  no    | 2 |            -|           -|
|          | NOT USED    |  no    | 3 |            -|           -|
|          | NOT USED    |  no    | 4 |            -|           -|
|          | NOT USED    |  no    | 5 |            -|           -|
|          | NOT USED    |  no    | 6 |        41512|       41521|
| hdla     | WHOLE DISK  |  no    | 7 |            0|       41980|
+----------+-------------+--------+---+-------------+------------+

41522 blocks for divisions, 459 blocks reserved for the system

 n[ame]       Name or rename a division.
 c[reate]     Create a new file system on this division.
 t[ype]       Select or change filesystem type on new filesystems.
 p[revent]    Prevent a new file system from being created on this division.
 s[tart]      Start a division on a different block.
 e[nd]        End a division on a different block.
 r[estore]    Restore the original division table.

Enter your choice or q to quit:
```

Each row in the **divvy** table corresponds to a filesystem (also known as a division).

You can divide the partition into as many filesystems as you like. Each filesystem must have the following:

- if the division does not already exist (NOT USED appears in the Type column) it must first be created using the **c** (create) command
- a beginning block number, defined by the **s** (start) command
- an ending block number, defined by the **e** (end) command
- a filesystem name, defined by the **n** (name) command. Filesystems can have any name you choose. For example, you could name a filesystem *u* (for "user"). This name in turn creates the device name (for example, */dev/u*).

WARNING Filesystem boundaries must not overlap. For example, filesystem 0 cannot end on the block number where filesystem 1 begins. Do not change the configuration of filesystem 7; it is reserved for internal use by the operating system.

Do not name a filesystem *usr*; this directory already exists on the *root* filesystem.

The default filesystem type is EAFS. If you wish to create filesystems of other types, you must use the **type** command.

Exit from **divvy** by entering **q**. The program prompts whether to install the new partition table, or return to the main menu. Select option **i** to install the partition table.

For more information, see the **divvy**(ADM) manual page.

27. The system now creates the filesystems on your hard disk. This takes several minutes. You see the following message:

```
Making filesystems
```

28. After creating the new filesystem(s), **mkdev hd** terminates. To make the filesystem(s) accessible, you must follow the instructions in the next section, "Adding the new filesystem(s)."

Adding the new filesystem(s)

Before leaving system maintenance mode, you must add the new filesystem to the system. To do this, follow these steps:

1. Enter the following command:

   mkdev fs

   Δ **sysadmsh** users select: Filesystems ⇨ Add

2. You see the following:

```
This program performs maintenance tasks required to add or delete
an existing filesystem. Would you like to:

         1. Add a new filesystem to system.
         2. Remove a filesystem.

Select an option or enter q to quit:
```

   Select **1**.

3. You are next prompted for the device name:

```
Enter a device name and press <Return> or q to quit:
```

   Enter the full pathname of the device from */dev*. The device name is derived from the name you gave the filesystem during the **divvy** phase. For example, to add a filesystem called *u*, you enter **/dev/u**.

4. You are now prompted to provide the name of the mount point to be used:

```
Enter a directory name and press (Return) or q to quit:
```

   This directory is where the filesystem is mounted. For example, a filesystem called *u* is mounted on the directory */u*.

5. The following is displayed:

```
Reserving slots in lost+found directory ...

When entering multiuser mode:
         1. Always mount filesystem
         2. Never mount filesystem
         3. Prompt before mounting filesystem

Select an option:
```

   If you want the filesystem mounted automatically at system startup, enter **1**. If you wish it mounted only by the request of the system administrator, select **2**. If you select **3**, the system prompts you at system startup whether or not you want the filesystem mounted.

6. You are then asked whether or not you want to permit users to mount filesystems:

```
Do you want to allow users to mount this file system? (y/n)
```

   You must respond **y** so that the system backup program can mount and unmount the filesystem as necessary.

7. The following messages are displayed when the process is complete.

```
Updating system files ...
Filesystem has been successfully added.
```

8. Next, you should mount the */x* filesystem using the following command:

   mount /dev/x /x

   Δ **sysadmsh** users select: Filesystems ⇨ Mount

Relinking the kernel

If you responded "no" when prompted to relink the kernel after installing a SCSI disk, you must run **link_unix** to rebuild the kernel manually with the new configuration information. Enter the following commands:

```
cd /etc/conf/cf.d
./link_unix
```

Δ **sysadmsh** users select: System ⇨ Configure ⇨ Kernel ⇨ Rebuild

Adding a CD-ROM drive

CD-ROM drives are supported using the ISO9660/High Sierra CD-ROM filesystem. A CD-ROM filesystem can be configured and mounted as a read-only filesystem under the UNIX system. The filesystem allows access to files which are described by the primary volume descriptor on the CD-ROM. Access to files described by secondary volume descriptors is not supported. At this time the system provides a level of support similar to that provided by the Microsoft CD-ROM Extensions for MS-DOS.

Configuring the drive

CD-ROM drives are added to the system as SCSI devices. You must specify the SCSI host adapter number, the host adapter type, Target ID, and logical unit number (LUN) as described in Figure 10-1 earlier in this chapter.

To add your CD-ROM drive to the system, follow these steps:

1. Bring the system down to maintenance mode. You can do this with the **su** option of the **shutdown**(ADM) command:

   ```
   shutdown su
   ```

2. Enter the following command:

   ```
   mkdev cdrom
   ```

3. You see:

```
CD-ROM Configuration Program

          1. Install a CD-ROM Drive
          2. Remove a CD-ROM Drive
          3. Install a CD-ROM/TAPE Driver
          4. Remove a CD-ROM/TAPE Driver

          Select an option or enter q to quit:
```

Enter 1 and press ⟨Return⟩. The following messages are displayed:

```
Updating system configuration

System files have been successfully updated
```

4. If the CD-ROM driver is not installed, you see:

```
The CD-ROM Driver must be configured before use.
Do you wish to configure the CD-ROM Driver now? (y/n)
```

   Enter y and press ⟨Return⟩.

5. You are prompted for the SCSI configuration data as follows:

```
Enter the prefix of the SCSI host adapter that supports this device
or press ⟨Return⟩ for the default: type
Enter h for a list of host adapters

Which type SCSI host adapter supports this device?
Select 0-n, or 'h' for help, or 'q' to quit:
```

   Enter the data requested, followed by ⟨Return⟩.

6. If you are adding a CD-ROM drive connected to a previously configured host adapter, skip to step 9. If you are configuring a secondary host adapter of a type already installed, you see the following:

```
hatype SCSI host adapter n is not configured.
Would you like to update the link-kit?
```

   The kernel already recognizes one of each type of host adapter; an additional entry must be created for the new one. Respond y and press ⟨Return⟩.

7. You may be prompted for one or more of the following, depending on your host adapter type (the system attempts to autoconfigure any values it can obtain from default files and other adapters installed of the same type):

```
What is the interrupt vector for this adapter?

What is the start IO address (hex) for this adapter?

What is the end IO address (hex) for this adapter?

What is the start controller memory address (hex) for this adapter?

What is the end controller memory address (hex) for this adapter?
```

The hardware appendix of the *Release Notes* contains a table of default SCSI host adapter configuration parameters. If you do not have a given type of adapter installed, you can probably use the values for another adapter. (You should check for conflicts before entering values; other hardware devices may already be using the same interrupt vector, I/O address, and so forth.) Enter each value as prompted and press ⟨Return⟩.

8. You are asked to confirm the values you supplied. Then the following prompt is displayed:

```
The following parameters will be used to configure hatype SCSI host adapter n

list of parameters

Update the link-kit?
```

If you need to make corrections, enter **n** and press ⟨Return⟩, then press ⟨Del⟩ to exit **mkdev hd** and start over. If the values are correct, enter **y** and press ⟨Return⟩ to add these values to the link kit.

9. Next, you see:

```
What is the Target ID for this device?
Select 0-7, or 'h' for help, or 'q' to quit:

What is the LUN of this device?
Select 0-7, or 'h' for help, or 'q' to quit:
```

Enter the information according to your configuration. Because the CD-ROM drive and its controller are one unit (referred to as "embedded") the LUN or logical unit number is simply 0.

10. Now that the data is entered, the program acknowledges the information and prompts for relinking of the kernel:

```
Updating SCSI configuration file...
The SCSI configuration file has been updated.

A new kernel must be built, to reflect the changes
to the SCSI configuration. Do you want to do this now? (y/n)
```

11. Unless you have installed a CD-ROM drive previously, respond **no** to this prompt. If this is your first CD-ROM drive, you will have to relink again later.

12. If this is the first time you have added a CD-ROM drive to the system, support for the CD-ROM filesystem must be added to the kernel using the **mkdev high-sierra** command. This command simply asks if you want to add or remove CD-ROM support. This time when you are asked if you want to relink the kernel, respond **yes.**

The CD-ROM drive is now ready for use.

CD-ROM device names

The CD-ROM driver supports both block and raw device access to the CD-ROM drive.

The device names for CD-ROM drive 0 are:

/dev/cd0	block device
/dev/rcd0	raw device

The manual page for **cdrom**(HW) contains more information about this device driver.

Accessing CD-ROM filesystems

CD-ROM filesystems are mounted with the **mount**(ADM) command. All utilities behave as expected. You can traverse the filesystem with the **cd** command, and so forth, including copying files from the CD-ROM filesystem to other filesystems with such utilities as **cp**(C) and **tar**(C).

All files on the CD-ROM appear to have access permissions of 555 (that is, files are readable and executable by all users and are not writable). Filesystems containing extended attribute records are supported, but the record format information and the file access permissions in the extended attribute record are not used.

NOTE To bring the CD-ROM drive online, you must insert a disk. If you attempt to bring up the drive without inserting a disk, the message `cannot open` is displayed.

Chapter 11

Maintaining system security

Every computer system needs protection from unauthorized people accessing the computer, disks, and system files. The security features present on your system represent enhancements to the basic security features of UNIX operating systems. The operating system is designed to meet the requirements of the C2 class of trust as defined by the Department of Defense's *Trusted Computer System Evaluation Criteria* (also known as TCSEC or the *Orange Book*).

This chapter explains how to use the security features to maintain a trusted system. Features affecting the ordinary user are described in the "Using a secure system" chapter of the *User's Guide*.

This chapter includes information on the following:

- an overview of system security
- running a trusted system
- protecting the data on your system
- terminal login management
- activity report generation
- detecting system tampering
- dealing with filesystem corruption
- daemon operations on a trusted system
- disabling C2 features

WARNING The security features of the operating system are useless if your hardware and media are not protected. You must protect the computer itself, the distribution diskettes, and any backup media from unauthorized access. This is accomplished by the following rules:

- Keep your system under lock and key when an operator is not present.
- Organize and lock up all backup media.
- Protect communication lines (UUCP, Ethernet, and terminals) from unauthorized access.

What is a trusted system?

Because there is no such thing as a computer system that is completely free from risk, systems are referred to as "trusted" rather than "secure". A trusted system is one that achieves a greater level of control over access to information, providing mechanisms to prevent (or at least detect) unauthorized access, along with additional means to confirm that these mechanisms are functioning properly. The C2 level of trust means that the system is designed to meet specific criteria in its security policy: accountability, assurance, testing, and documentation.

The security features of your trusted system are an extension of features present on most UNIX systems. Full compatibility with existing UNIX system mechanisms is maintained while expanding the protection of user and system information. A large part of system administration involves maintaining and protecting system information as described in this chapter.

At installation time, you were asked to select the security defaults to be used on your system. In addition, you can customize any of the defaults to the needs at your site.

As administrator, your actions are crucial to maintaining a trusted system. Any lapses from a trusted state invite system penetrations. To be effective in your administrative position, you must understand the system's security policy, how it is controlled by system information (databases), and how changes you make affect user and administrator actions.

Trusted system concepts

The following section defines the basic concepts of a trusted system. As administrator, you must understand these concepts and know where security-relevant information is kept to run the system properly. This section only introduces these topics; later sections in this chapter provide further details and describe maintenance procedures.

Trusted computing base

A collection of software called the Trusted Computing Base (TCB) maintains the parts of the system that are related to security. The TCB consists of the UNIX system kernel (the heart of the operating system) and the trusted utilities that reference and maintain relevant security data. The TCB implements the security policy of the system. The security policy is a set of rules that oversee and guard interactions between "subjects" (such as processes, which are programs running on the system) and "objects" (such as files, devices, and interprocess communication objects). At the C2 level, this consists of Discretionary Access Control (DAC), discussed later in this section, and object reuse, the latter of which dictates that information in a storage object must be cleared before allocation. Much of the software that you interact with is part of the system's TCB. The **sysadmsh**(ADM) provides a menu-driven, administrative interface to help you maintain the TCB.

Accountability

An action is "accountable" if it can be traced to an individual person. On a trusted system, all actions can be traced to a responsible person. Most UNIX systems lack good accountability because some actions cannot be traced to a person. For example, pseudo-user accounts, such as **lp** or **cron**, run anonymously; their actions can be discovered only by changes to system information. As described later, a trusted UNIX system improves accountability by associating each account with a real user, auditing every action, and associating each action with a specific user on the system.

On a typical UNIX system, each process has a real and effective user ID as well as a real and effective group ID. A process with the effective user ID set to *root* can set these identifiers to any user. The C2 level of trust requires that the TCB be able to identify each user uniquely and thus enforce individual accountability. The concept of user identity is expanded on trusted UNIX systems to add a separate identifier called the *login user identifier* (LUID). The LUID is an indelible stamp on every process associated with a user. The LUID identifies the user who is responsible for the process's session. Once stamped, the process's LUID cannot be changed by anyone. Child processes inherit the LUID of their parent.

Discretionary access control

Discretionary Access Control (DAC) determines whether a user has access to desired data. That information is within an"object" (file, device, and so on) that the user's process is trying to use. On most UNIX systems, object protection is enforced through the relationship between the user and the group of a process and the owner, group and other mode bits of the object. The protection attributes of these objects are at the discretion of the object's owner, who can change the protection bits on a file and even give the file away (change ownership). A trusted UNIX system extends the standard discretionary access control rules used by the UNIX file permissions by restricting the following:

- ability to set the SUID and SGID (set user or group ID on execution) bit on files
- ability to change ownership of files (with **chown**(C))
- potential misuse of SUID, SGID, and "sticky" permissions by clearing these bits whenever a file is written

Authorizations

An authorization is a user attribute that is required to perform certain actions. Most UNIX systems make all access decisions based on the simple file permissions or on whether the process making the access is owned by *root*. The *root* account can perform system actions that no other process can. The TCB defines two types of authorizations: kernel and subsystem. Kernel authorizations are associated with processes. They allow a process to perform certain actions if the process has the requisite privilege. Subsystem authorizations are associated with users. They allow the user to perform a special action using a subsystem's commands (trusted utilities and programs). A "subsystem" is a related collection of files, devices, and commands that serve a particular function. For example, the *lp* subsystem consists of the print spooler files, the printer devices, and commands such as **lpadmin**(ADM) that help maintain the subsystem.

Kernel authorizations are stored in an "authorization set" associated with every process. The authorization set is a list of privileges that allow a type of action if the authorization is present, and do not allow the action if the authorization is absent. Authorizations are set either by the system defaults, or are defined for a specific user.

Identification and authentication (I&A)

When a user logs into a non-trusted UNIX system, limited identification and authentication (I&A) takes place. The system searches the password database (*/etc/passwd*) for the user name. If the user name is found, the system authenticates the user by comparing the password entered to the encrypted version of the password in the user's password database entry. Some rules concerning the characteristics of the password and the ability to change it may be enforced, but these rules have been shown to be insufficient to guard against penetration.

A trusted system extends the standard UNIX system I&A mechanisms. There are more rules enforcing the types of passwords that can be used. There are new procedures for generating and changing passwords. The location and protection of certain parts of the password database differs from that of other UNIX systems. The administrator also has greater control over the login process. A separate role, called authentication (or accounts) administrator (subsystem authorization *auth*), maintains this aspect of the system. This administrator's responsibilities are described in detail in later sections.

Auditing

Most UNIX systems keep a limited record of system actions with their accounting subsystem. The accounting subsystem writes a single accounting record upon completion of each user process. The trusted operating system provides an extensive series of records, or "trail," of actions. In this trail is a record of every access between subject and object (successful and unsuccessful) and every change of subject, object, and system characteristics. The audit subsystem is controlled by a separate role called audit administrator (subsystem authorization *audit*). The audit administrator decides how much information is recorded, and how reliably it is recorded, and maintains the information once it is collected. The audit subsystem provides the audit administrator with an extensive history of system actions. This helps the administrator to identify what happened to the system, when it occurred, and who was involved.

Protected subsystems

UNIX systems provide the set user ID (SUID) and set group ID (SGID) mechanisms. (The ability to set user or group ID on execution is accomplished via the **setuid** and **setgid** system calls and via the **setuid** and **setgid** permission bits on files. For more information, see the **chown**(C), **setuid**(S), and **setgid**(S) manual pages.) With these you can construct programs maintaining private information. This information can only be accessed or modified by the operations implemented in the programs. The TCB defines several protected subsystems. Each of these subsystems consists of a collection of private information (files and/or databases), any related devices, and the utilities and commands used to maintain that information. The protected subsystems use the SUID/SGID mechanisms to protect their private files, databases, and devices from unrestricted access. The trusted system extends the notion of a protected subsystem in several ways:

- It provides more precise control of users and groups who maintain certain collections of system resources (private information).
- It keeps a separate database of users allowed to run the programs that maintain the private information.

- It does not require users to log in as the subsystem administrator but rather uses the database to check the subsystem authorization. This satisfies the full accountability requirement for all actions performed by subsystem programs. (If users log in to anonymous accounts to perform system administration, there is no way to determine who performed a given action.)

Running a trusted system

You have already chosen the security scheme (Low, Traditional, Improved, or High) to be used on your system. Even if you did not choose to run a trusted (High) system, you should consider the following options, which are useful under any set of defaults:

- assigning a single person to administer the system or a group, with each person assigned an individual subsystem
- assigning kernel authorizations to users requiring additional privileges
- deciding how strictly you wish to control and monitor access to your system
- automatically logging out idle users
- deciding how to use the auditing features

Assigning administrative roles

The first basic choice you must make is who will maintain the trusted system. You can have a single, all-powerful super user with the *root* login, or you can assign parts of the administrative responsibility to other users, assigning no more power than is necessary to administer a single aspect of system operation. The administrative tasks for a trusted UNIX system are split into a number of logical roles. Each role is responsible for maintaining one aspect of the system. The idea of specific administrative roles (and their associated tasks and responsibilities) is pivotal to your understanding of a trusted operating system. Each logical role can be assigned to the same person or to separate members of an administrative staff. Each extended role has a special authorization and a **sysadmsh** selection. That association, together with a sophisticated tracking system, enables the administrator to maintain a clear record of administrative actions. This helps to prevent problems and makes existing problems easier to identify and solve.

To perform the tasks associated with an administrative role, an administrator must have the appropriate subsystem authorization. Table 11.1 lists the subsystems, associated authorizations, and the areas of the system maintained by each role.

Table 11-1 Protected subsystems and administrative roles

Role	Subsystem Authorization	Area
System Administrator	su	su access to other accounts
	sysadmin	*not implemented*
Audit Administrator	audit	Audit databases and audit trail
Accounts Administrator	auth	System Accounts
Operator	backup	Filesystem backups
Cron Administrator	cron	at and cron subsystem
Printer Administrator	lp	Line printer subsystem
*	mem	Access to process table data
*	terminal	Terminal device permissions

* These are not administrative roles, but are listed for completeness.

It is vital that you understand the responsibilities for each role and the impact of your actions on the security of the system. You should configure and run the system based on the sensitivity of information kept on your site, the perceived degree of cooperation and expertise of your users, and the threat of penetration or misuse from insiders and outsiders. Only your vigilance and proper use of the system can keep the system trusted and protect the integrity of your system.

To assign a subsystem authorization, make the following **sysadmsh** selection:

Accounts ➪ User ➪ Examine:Privileges

NOTE You might notice that each primary subsystem authorization appears to be identical to the group name for that subsystem. This means that if a user is a member of a subsystem group, there is an implied ability to access the files of that subsystem. You should never make a user a member of a subsystem's group, as this can put actual data files at risk. Use the proper subsystem authorization to permit access to the subsystem.

Administering subsystems with sysadmsh

Certain subsystems are logical divisions rather than actual areas of system administration. For example, the *mem* authorization is not associated with an administrative role, but it controls access to kernel memory structures. Other subsystems require administration and have **sysadmsh**(ADM) selections. These subsystems can be assigned to individuals, and documentation is provided for each area. Table 11.2 lists each of the subsystems that must be administered, their **sysadmsh** selections and the chapters that deal with them.

Table 11-2 Subsystems and sysadmsh selections

Protected Subsystem	Purpose and Role	sysadmsh Selection	Chapter or Appendix of This Guide
Line Printer	Printer administration	Printers	"Using printers"
Backup	Filesystem copies	Backups	"Backing up filesystems"
Authentication	Account maintenance	Accounts	"Administering user accounts"
Cron	Task scheduling	Jobs	"Authorizing the use of job scheduling commands"
Audit	Auditing	System ➪ Audit	"Using the audit subsystem"

The subsystems are described in detail in **subsystem**(M). This page lists all the programs and data files associated with a subsystem. Most of the functions normally exercised by the super user on non-trusted UNIX systems are delegated to the protected subsystems detailed in this section. However, some functions still need to be performed by the super user. This includes mounting and unmounting filesystems, and traversing the entire file tree. Only the super user can do everything. Restrict the *root* password to a few users and assign a responsible user to the *root* account. (See the "Administering user accounts" chapter of this guide.)

Assigning kernel authorizations

As discussed previously, the TCB has two types of authorizations: kernel and subsystem. Table 11.3 contains a list of kernel authorizations.

Table 11-3 Kernel authorizations

Authorization	Action
configaudit	ability to modify audit parameters
writeaudit	ability to write audit records to the audit trail
execsuid	ability to run SUID programs
chmodsugid	ability to set the SUID and SGID bit on files
chown	ability to change the owner of an object
suspendaudit	suspends the audit of a process

Most users require only **execsuid** kernel authorizations to perform routine tasks. If the user needs to create files with the SUID or SGID bits, they must have the **chmodsugid** authorization. To change ownership of a file (give it away), the **chown** authorization is required. If a user does not have this authorization, ownership of files can only be changed by *root*. The audit kernel authorizations (**configaudit**, **writeaudit**, and **suspendaudit**) should never be assigned to anyone other than the audit administrator. They are intended for use by a program designed to run as a trusted application.

NOTE Restricted **chown** is required for NIST FIPS 151-1 conformance. The **chown** authorization should not be assigned to users if you wish to conform to NIST FIPS 151-1 requirements.

To assign a kernel authorization, make the following **sysadmsh** selection:

Accounts ⇨ User ⇨ Examine:Privileges

Users assigned administrative roles must also have certain kernel authorizations to perform the tasks required by the subsystem. The requisite kernel authorizations are shown in Table 11.4.

Table 11-4 Subsystem kernel authorization requirements

Subsystem	Required kernel authorization
audit	configaudit, writeaudit, execsuid
auth	chown, execsuid
backup	execsuid
lp	chown
cron	execsuid, chown, chmodsugid
sysadmin	execsuid, chown, chmodsugid

Controlling system access

One important aspect of operation on a trusted system is locating potential problems relating to security. The restriction mechanisms fall into three categories, all of which can be customized and reported on:

- password restrictions
- terminal use restrictions
- login restrictions

Password restrictions

The Department of Defense's *Password Management Guideline* (also known as the *Green Book*) was used as a model for password restrictions, and users are subject to much stricter password checking than traditional UNIX systems. The authentication administrator can either allow users to pick their own passwords or have the system generate passwords for them. When chosen, the password can be subjected to simple or extensive checking for obviousness, again at the option of the authentication administrator.

The lifetime of a password is defined as follows:

- *The password is valid.*
- *The password has expired.* The user can still log in and change the password (if authorized to do so).
- *The password is dead.* The user is locked out and the administrator must unlock the account.

If users are not allowed to change passwords, a new password must be assigned. Users must notify the administrator when their account is locked; there is no notification mechanism other than use of regular reports on impending expirations as described in "Activity report generation" in the "Administering user accounts" chapter of this guide, or use the **prwarn**(C) utility.

A popular tactic among users on systems where periodic password changes are enforced is to change their password once, thus satisfying the requirement, then simply change their password back again to the one they used before. To prevent a user from doing this, the authentication administrator can also set a minimum change time on a password, before which a user may not change passwords. All of these parameters can be changed on a system-wide (System Defaults database) and per-user (Protected Password database) basis. See "Changing default password restrictions" and "Changing a user password or password parameters" in the "Administering user accounts" chapter of this guide.

By default, the user account initialization files (*.cshrc*, *.profile*, and so forth) call the **prwarn**(C) utility to warn users of impending password expiration and prevent their accounts from being locked. Expirations can be an annoying occurrence if a system administrator is unavailable. If your system is not attended by administrators on a daily basis, you might want to extend the password lifetime parameter accordingly.

Terminal use restrictions

Terminals are gateways to the system. In addition to the use of account passwords, terminals can be protected from attempts to penetrate the system. You can define the maximum number of failed login attempts, which is typically associated with attempts to crack an account password. Terminals that exceed the maximum permissible number of attempts will be locked and you, the accounts administrator, must unlock them. In addition, you can specify an interval that must elapse between login attempts, which can further thwart attempts to break a password. (Similar restrictions can be defined for accounts rather than terminals.) To change or examine terminal restrictions, refer to the "Terminal login management" section later in this chapter.

Login restrictions

As with terminals, user accounts have parameters associated with the number of login attempts and retry intervals. To change or examine login restrictions, refer to "Changing default login restrictions" in the "Administering user accounts" chapter of this guide.

Status reporting on access restrictions

Each of the restrictions discussed in this section have reporting facilities. For example, you can generate a report on the login records for a terminal or group of terminals, or report on user accounts with passwords that are about to expire. The procedures for running these reports are found in "Activity report generation" later in this chapter.

Logging out idle users

Finding a user logged into the system who has not entered any command or information for a long time can indicate that the user left the terminal and forgot to log out. The **idleout**(ADM) command monitors line activity and logs out any user whose terminal remains idle longer than a specified period of time. You must be logged in as the super user to run **idleout**.

To begin monitoring line activity for the system, enter:

idleout

Δ **sysadmsh** users select: System ➪ Configure ➪ Autologout

The **IDLETIME** variable in the */etc/default/idleout* file determines how long a user's terminal can remain idle before the system logs the user out. If the value of **IDLETIME** contains a colon (:), **idleout** calculates the time in hours; otherwise, **idleout** calculates the time in minutes.

You can also specify the acceptable idle time on the command line in the one of the following forms:

idleout ***minutes***

or

idleout ***hours:minutes***

If you want **idleout** to run automatically when you reboot your system, enter the command name, **idleout**, on a line by itself in the file */etc/rc2.d/S88USRDEFINE*.

Using auditing on your system

Auditing keeps detailed records of system usage, enabling you to determine if any tampering has occurred (whether attempted or successful). However, auditing can require additional supervision and disk space, depending on how long it is enabled. Auditing is discussed extensively in the "Using the audit subsystem" chapter in this guide.

NOTE It is not necessary to keep auditing enabled. It can be a useful tool if tampering is suspected. Because most systems calls are recorded when auditing is enabled, it is also an excellent debugging tool for programs.

Protecting the data on your system

The primary data protection on your system is the use of standard UNIX system permissions on files and directories. If you are unfamiliar with file permissions, you should read the *Tutorial*. Understanding the permission bits that you can set to protect files and directories is crucial to the security of your system. The default permissions for files created on your system are governed by the system-wide **umask**(C), which can also be customized by individual users.

Your system also includes important filesystem features that extend the protection of standard UNIX systems. These features greatly enhance the security of the system. One of them, SUID and SGID bit clearing upon file writes, is passive in that it requires no action by the system administrator. Other features are active, meaning that you can select them for particular objects. These active features, discussed below, include the special use of the sticky bit on directories, data encryption, and precautions to follow when importing data files from another system.

SUID/SGID and sticky bit clearing on writes

SUID and SGID permission bits on files change the user and group IDs of a process on execution of a program. Ordinary users should not be able to set these bits, and their use is restricted by the **chmodsugid** kernel authorization. (See "Assigning kernel authorizations" in this chapter.) Trusted UNIX guarantees that the SUID and SGID bits are cleared on files that are written. The reason for the clearing is to prevent a user from substituting another program to take advantage of its SUID or SGID bits, which they could not otherwise set.

NOTE The clearing of SUID/SGID bits can be disabled if desired. See the "Disabling C2 features" section later in this chapter for details.

An SUID bit shows as an "s" in the permissions of a file. In Example 11-1, the bit clearing is demonstrated twice (user input is in boldface).

Example 11-1 Bit clearing examples

```
$ id
uid=76(blf) gid=11(guru)
$ ls -l myprogram
-rwsrwsrwt   1 root  bin    10240 Jan 11 22:45 myprogram
$ cat sneakyprog > myprogram
$ ls -l myprogram
-rwxrwxrwx   1 root  bin    10240 Mar 18 14:18 myprogram
$ ls -l anotherprog
-rws------   1 blf   guru  83706 Dec 15  1987 anotherprog
$ strip anotherprog
$ ls -l anotherprog
-rwx------   1 blf   guru  17500 Mar 18 14:19 anotherprog
```

In the example, user *blf* (the **id**(C) utility was used to show the identity) first uses the **cat** utility to replace the contents of the file *myprogram*. The SUID bit is removed during this process. The second example demonstrates that the bit clearing is even done on files owned by the same user. When *blf* strips the file (removing the debugging information in a compiled binary file), the SUID bit is also removed. You should be aware that the clearing happens when files are replaced. Adjust any installation scripts to reset the proper modes. With this feature, you can place these bits on user programs without fear that the user can switch programs in the same file.

NOTE SUID and SGID do not work on shell scripts.

The SUID, SGID, and sticky bits are not cleared on directories. The SUID bit has no meaning for directories, while both the SGID and sticky bits have a meaning for directories that warrant their remaining there. This is described next.

The sticky bit and directories

Another important enhancement involves the use of the sticky bit on directories. A directory with the sticky bit set means that only the file owner and the super user may remove files from that directory. Other users are denied the right to remove files irrespective of the directory permissions. Only the super user can place the sticky bit on a directory. Unlike with files, the sticky bit on directories remains there until the directory owner or super user explicitly removes the directory or applies **chmod**(C) or **chmod**(S) to it. Note that the owner can remove the sticky bit, but cannot set it.

You can gain the most security from this feature by placing the sticky bit on all public directories. These directories are writable by any non-administrator. You should train users that the sticky bit, together with the default **umask** of 077, solves a big problem area of less secure systems. Together, both features prevent other users from altering or replacing any file you have in a public directory. The only information they can gain from the file is its name and attributes.

Example 11-2 illustrates the power of such a scheme. The sticky bit is the "t" in the permissions for the directory. (On UNIX systems, the present directory is shown in a file listing as a dot " . ", and two dots " .. " represent the directory level above the present one.)

Example 11-2 Sticky bit example

```
$ iduid=76(slm) gid=11(guru)
$ ls -al /tmp
total 64
drwxrwxrwt   2 bin      bin       1088 Mar 18 21:10 .
dr-xr-xr-x  19 bin      bin        608 Mar 18 11:50 ..
-rw-------   1 blf      guru     19456 Mar 18 21:18 Ex16566
-rw-------   1 blf      guru     10240 Mar 18 21:18 Rx16566
-rwxr-xr-x   1 slm      guru     19587 Mar 17 19:41 mine
-rw-------   1 slm      guru       279 Mar 17 19:41 mytemp
-rw-rw-rw-   1 root     sys         35 Mar 16 12:27 openfile
-rw-------   1 root     root        32 Mar 10 10:26 protfile
$ rm /tmp/Ex16566
rm: /tmp/Ex16566 not removed.  Permission denied
$ rm /tmp/protfile
rm: /tmp/protfile not removed.  Permission denied
$ cat /tmp/openfile
      Ha! Ha!
You can't remove me.
$ rm /tmp/openfile
rm: /tmp/openfile not removed.  Permission denied
$ rm -f /tmp/openfile
$ rm /tmp/mine /tmp/mytemp
$ ls -l /tmp
drwxrwxrwt   2 bin      bin       1088 Mar 18 21:19 .
dr-xr-xr-x  19 bin      bin        608 Mar 18 11:50 ..
-rw-------   1 blf      guru     19456 Mar 18 21:18 Ex16566
-rw-------   1 blf      guru     10240 Mar 18 21:18 Rx16566
-rw-rw-rw-   1 root     sys         35 Mar 16 12:27 openfile
-rw-------   1 root     root        32 Mar 10 10:26 protfile
$ cp /dev/null /tmp/openfile
$ cat /tmp/openfile
$ cp /dev/null /tmp/protfile
cp: cannot create /tmp/protfile
$ ls -l /tmp
drwxrwxrwt   2 bin      bin      1088 Mar 18 21:19 .
dr-xr-xr-x  19 bin      bin       608 Mar 18 11:50 ..
-rw-------   1 blf      guru    19456 Mar 18 21:18 Ex16566
-rw-------   1 blf      guru    10240 Mar 18 21:18 Rx16566
-rw-rw-rw-   1 root     sys         0 Mar 18 21:19 openfile
-rw-------   1 root     root       32 Mar 10 10:26 protfile
$
```

The only files removed are those owned by user *slm* (the user in the example). The user *slm* could not remove any other file, even the accessible file */tmp/openfile*. However, the mode setting of the file itself allowed slm to destroy the file contents; this is why the **umask** setting is important in protecting data. Conversely, the mode on */tmp/protfile*, together with the sticky bit on */tmp*, makes */tmp/protfile* impenetrable.

All public directories should have the sticky bit set. These include, but are not limited to, the following:

- */tmp*
- */usr/tmp*
- */usr/spool/uucppublic*

If you are unsure, it is far better to set the sticky bit on a directory than to leave it off. You can set the sticky bit on a directory with the following command, where ***directory*** is the name of the directory:

chmod u+t *directory*

Using data encryption

Data encryption can also be used to enhance the security of your system through the **crypt**(C) command. These features are described in "Using a secure system" in the *User's Guide*.

NOTE The data encryption software is not included in your distribution, but is available by request only within the United States. You can request this software from your dealer or distributor.

Importing data

Files and filesystems brought into the system from elsewhere are a threat to the system if not handled properly. This section discusses techniques to use when importing files to your system.

Files

Do not take for granted the permissions on an imported file. Not only are the */etc/passwd* and */etc/group* files different on each system, but the policies on differing systems dictate setting different modes. These considerations are critical when the imported files are system files.

To minimize your intervention and clean up after importing files, train everyone on the system to use archive program options that do not reset ownerships. The files are owned by the user importing the files. The **cpio**(C) program only changes the ownerships of files when the invoking user is the super user. The archive programs generally reset the file modes to those described on the media containing the archive. In addition to having a mode that is more permissive than necessary, files can have SUID, SGID, or sticky bits set. All of these situations can create security problems for you.

To minimize the effects of the archive permissions, use archive options that examine the contents without extracting anything. For example, the **-tv** option to **tar** and the **-tv** option to **cpio** let you see the modes of the files on tape and prepare for any ill effects when extracting files. When bringing in unfamiliar archives, first import files into a hierarchy not accessible to others. Then manually move the files, after adjusting the ownership and modes according to your system policy.

Filesystems

Mounting filesystems that were created or handled elsewhere has all the same concerns as importing files. Filesystems also bring with them two extra concerns. The first is that the filesystem may be corrupted. The second is that file permissions on the filesystem may not be acceptable for your system. In either case, mounting a bad filesystem can cause the system to crash, the data on the imported file system to be further corrupted, or for other filesystems to go bad from side-effects. This is why the **mount**(ADM) command is reserved for the super user. The **fsck**(ADM) program should be run on all filesystems before they are mounted. If the filesystem contains system files, the **integrity**(ADM) and **fixmog**(ADM) utilities should also be run after it is mounted.

Imported filesystems can contain file permissions not suitable for your system. The super user of the imported filesystem may have set ownerships, sticky bits, special files, SUID/SGID bits, and file tree compositions incompatible with your system policies. Special files may exist with different ownerships and modes that you cannot allow.

You can use the **-s** option to **ncheck**(ADM) to locate any potentially dangerous SUID files before mounting. Filesystems, like files, should be scanned before they are mounted. The first time a filesystem is mounted in your control, it is best to mount it in a private directory so you may scan the filesystem manually before mounting it in its normal place. Examine the file organization, the owners and modes of the files, and the expected use of the filesystem.

Terminal login management

The Terminal Control database stores parameters about system terminals. This database gives the administrator control over how many unsuccessful login attempts can be made before the terminal locks. It also stores the login activity for the terminal. When you install a terminal or printer, the information is automatically added to the Terminal Control database. However, you must modify these entries to govern how they can be used and what security procedures will be observed.

The Terminals selection of the "Accounts" menu has the following selections:

Examine	views or modifies an existing terminal entry
Create	makes a new terminal entry
Delete	deletes an existing terminal entry
Lock	locks a specific terminal
Unlock	unlocks a specific terminal
Assign	manages device name equivalencies database

Basic entries in the Terminal Control database are automatically created as tty devices are added to the system. The selections that you use most of the time are Examine, Lock, and Unlock. By default, the system manages entries as required.

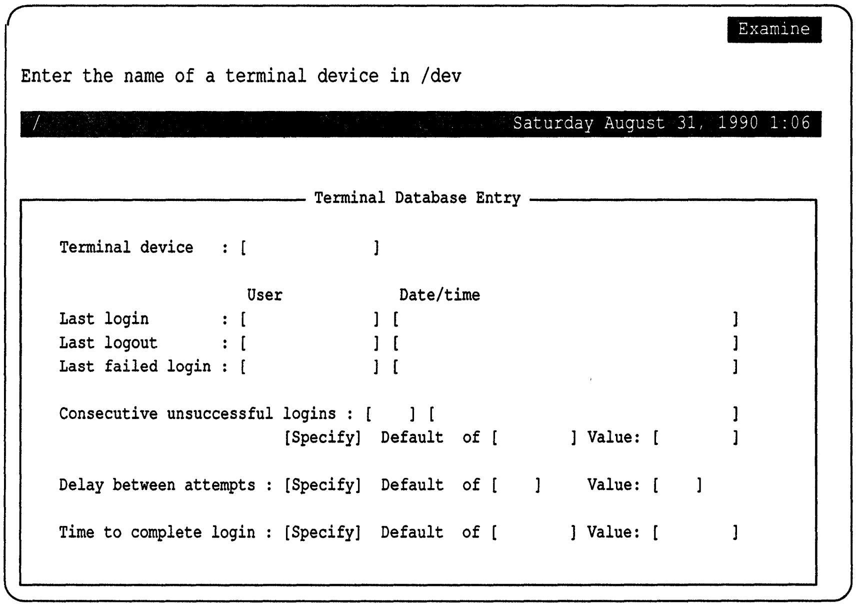
The remaining selections, Create, Delete, and Assign are special cases that are used when special hardware or software has been added to the system that requires manual configuration.

Examining a terminal entry

To modify settings for a terminal, select the following:

Accounts ➪ Terminal ➪ Examine

The following screen is displayed:



```
                                                                      Examine

Enter the name of a terminal device in /dev

/                                             Saturday August 31, 1990 1:06

                          Terminal Database Entry

   Terminal device   : [              ]

                         User              Date/time
   Last login        : [              ] [                                    ]
   Last logout       : [              ] [                                    ]
   Last failed login : [              ] [                                    ]

   Consecutive unsuccessful logins : [    ] [                                ]
                        [Specify]  Default  of [       ] Value: [        ]

   Delay between attempts : [Specify]  Default  of [    ]   Value: [    ]

   Time to complete login : [Specify]  Default  of [       ] Value: [        ]
```

This screen lets you examine the current status of the terminal. All choices require a terminal name, which is the directory entry for the terminal in the */dev* directory. A value of "INFINITE" for the "Consecutive unsuccessful logins" disables this type of lock for the terminal (abbreviations are acceptable). The Terminal Control entry is related to the device assignment database, as described later in this chapter.

NOTE The super user can break the terminal lock on the system console. This is to avoid a complete lock-out of all users everywhere. Because this special login is allowed, you should physically protect the system console.

Redefining login attempt limit

If the login restriction on a terminal proves problematic, or has proven too loose, use the following **sysadmsh** selections to define these limits:

Accounts ➪ Terminal ➪ Examine

See the previous section for the form displayed. Change the "Consecutive unsuccessful logins" and "Delay between attempts" as desired.

Locking or unlocking a terminal

To lock and unlock a terminal, respectively, use the following **sysadmsh** selections:

Accounts ➪ Terminals ➪ Lock

Accounts ➪ Terminals ➪ Unlock

When the prompt appears for the terminal, enter the name, for example: `tty01`. When a terminal is locked, the following message is displayed when an attempt is made to log in:

```
Terminal is disabled -- see Account Administrator
```

Setting up device equivalencies database

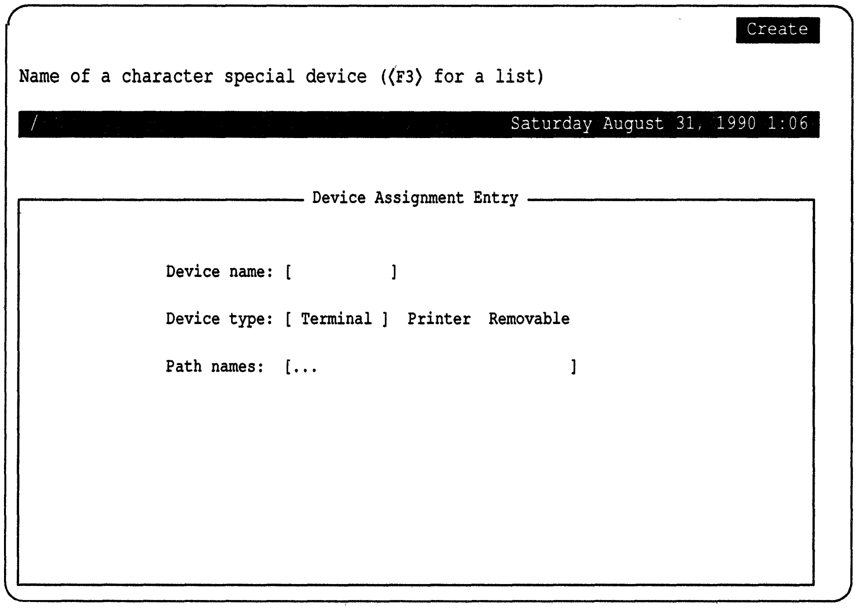
The purpose of the device assignment database is to record terminal devices that are physically the same, but referred to by different pathnames (they are linked, or are the same device with and without modem control, and so on). This equivalency mapping is very important in the case of terminals, where it ensures that the login history and terminal locking applies correctly whichever device pathname the system happens to see.

One example is someone disabling *tty1a* and then enabling *tty1A*. Because the device assignment database records the equivalence of these devices, the unsuccessful login count, for instance, is maintained. Again, the system does this automatically with devices that it recognizes by default. Any special device nodes created for unusual hardware or software have to be configured and added manually. You should only do this if the documentation has instructed you to do this or you know what you are doing.

To change a device assignment entry, select the following:

Accounts ➪ Terminals ➪ Assign ➪ Create

The following form is displayed:



```
                                                                  Create

Name of a character special device (<F3> for a list)

/                                                 Saturday August 31, 1990 1:06

 ---------------------------- Device Assignment Entry ----------------------------


                  Device name: [          ]

                  Device type: [ Terminal ]  Printer  Removable

                  Path names:  [...                         ]
```

Enter the device name found in */dev*. Then select the type of device, whether terminal, printer, or removable device such as a hard disk cartridge. You should then include the full pathnames of any links to the device.

Activity report generation

It is possible to create reports on the status of three important aspects of system operation:

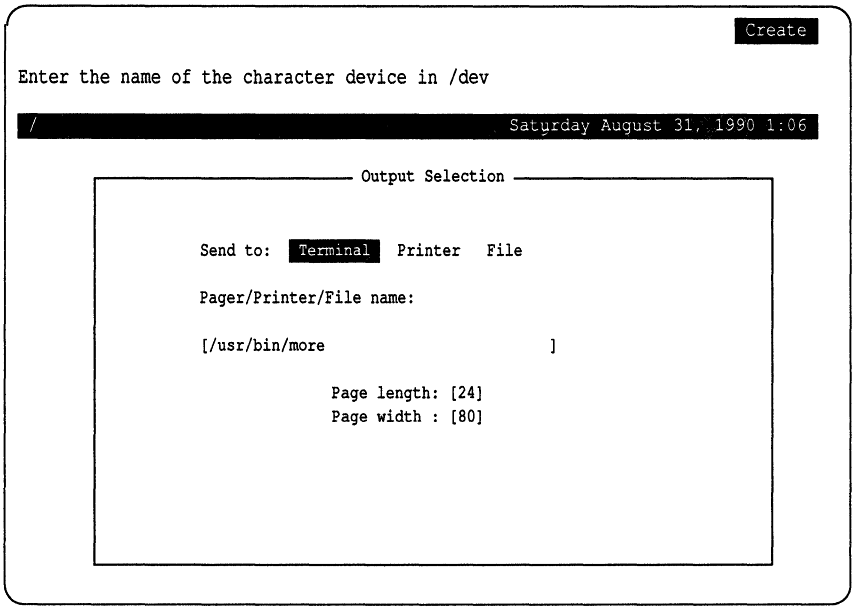
Passwords	reports on accounts by password status.
Terminal	reports on access by terminal status.
Login	reports on login activity by user, group, or terminal.

You can use the reports for security purposes (for example, listing parameters in the Protected Password and Terminal Control databases). Because these reports show system and peripheral usage, you may find them useful to fine-tune and reconfigure the system.

For all the reports, upon executing the screen you are asked to direct the output to the screen, the printer or a file.

You can filter screen output through any of the system pagination programs. The program defined by the **PAGER** environment is set up as the default; if **PAGER** is not defined, the **more**(C) program is used. For printer output, you can name the printer device; if you do not name it, the system default printer destination is used. If redirecting output to a file, use full pathnames. No matter what category of report you select, you are always requested to define how you want the data displayed: on screen, to a printer, or into a file.

The output screen looks like this:



```
                                                                    Create

Enter the name of the character device in /dev

 /                                               Saturday August 31, 1990 1:06

               ─────────────────── Output Selection ──────────────────

                         Send to:  Terminal  Printer   File

                         Pager/Printer/File name:

                         [/usr/bin/more                       ]

                                  Page length: [24]
                                  Page width : [80]
```

Reporting password status

To generate reports based on password status, make the following **sysadmsh** selection:

Accounts ➪ Report ➪ Password

Password status can be reported in several categories:

Impending	reports on accounts with passwords about to expire
Expired	reports on accounts with expired passwords
Dead	reports on accounts with dead passwords
User	reports on a single user
Group	reports on a single group of users
Full	lists all entries in password database

NOTE The default account configuration files (*.cshrc*, *.profile*, *.kshrc*, and so forth) automatically execute the **prwarn**(C) utility at login time to warn users about impending password expiration.

The Impending option reports on accounts that have, or will soon have, expired passwords. This includes all accounts with already-expired passwords as well as those that will expire within one week. Although an impending expiration is not actually an error, this report lets you see users who wait until the last moment to change passwords. You may want to revise the system-wide and per-user password expiration periods based on information obtained here.

The Expired option reports on all accounts with expired passwords. These may or may not be dead passwords. All such accounts need some administrative action before the account is usable; minimally, the password must be changed.

The Dead option reports on those accounts whose password lifetime has expired, which causes the account to reject further logins.

The User option reports on the individual user that you specify. Enter the user's login name to activate it.

The Group option reports on a single specified group. This report includes all the users who belong to the specified group.

Finally, the Full option reports statistics for all users on the system.

The reports use the following abbreviations:

Dflt	Default.
Y,N,D	Yes, No, Default. Some selections have three possible values: yes, no, and the default value used by the system, which can be either yes or no.

Example report: group

Example 11-3 is a sample report on the password activity of group "hamster." The abbreviations under "Password Parameters" correspond to the system-wide default password parameters.

Example 11-3 Sample password database report by group

```
                       Password Database Report
                            System  unix
                       Wed Mar 22 10:56:29 1991

                          Password Parameters
[1] User Name Type     Min  Exp  Life Rnd? Pck? Rst? Lck?
    ---- ---- ----     ---  ---  ---- ---- ---- ---- ----
          Last Changes               Last Logins       Consec
[2] Success     Failed      Success     Failed      #Failed
    -------     ------      -------     ------      -------

[3] Kernel Authorizations
    ------ --------------
[1] alvin     general  Dflt Dflt Dflt D    D    D    Y
[2] 05/22/90    NEVER       05/22/90    NEVER        -
[3] DEFAULT
[1] simon     general  Dflt Dflt Dflt D    D    D    N
[2] 05/22/90    NEVER       05/22/90    NEVER        -
[3] DEFAULT
[1] theodore  general  Dflt Dflt Dflt D    D    D    N
[2] 05/22/90    NEVER       05/22/90    05/22/90     -
[3] DEFAULT
```

Reporting terminal activity

To generate reports based on terminal activity, make the following **sysadmsh** selection:

Accounts ⇨ Report ⇨ Terminal

This allows you to get statistics on the Terminal Control database. The report contains any lock conditions, unsuccessful attempts to log in at the terminal, and the delay between login attempts. Similar to the Password reporting, you can select the report to apply to a single terminal, a range of terminals, or all terminals.

When you select a user or group, the report includes both the last successful and the last unsuccessful login. The number of unsuccessful attempts is also reported. As this number approaches the maximum login tries for the account, you should determine the cause for the problem. Most accounts should show a low number of login attempts.

When you select one or more terminals, the report includes the last successful, last unsuccessful, and last logouts on the terminal. The number of unsuccessful attempts on this terminal is also reported. Both report types can provide you with valuable data on how the system is being used.

Example 11-4 is a sample output of a terminal report.

Example 11-4 Sample terminal report

```
                 Terminal Database Report
                       System  unix
                 Wed Mar 22 10:58:42 1991

           Admin Login  Unsucc  Max Unsuc
Tty  Name Lck?  Delay Attempts Attempts
---  ---- ----  ----- -------- --------
console    D    Dflt  2        Dflt
tty02      D    Dflt  None     Dflt
tty03      D    Dflt  None     Dflt
tty04      D    Dflt  None     Dflt
tty05      D    Dflt  1        Dflt
tty06      D    Dflt  None     Dflt
tty07      D    Dflt  None     Dflt
tty08      D    Dflt  None     Dflt
tty09      N    Dflt  None     Dflt
tty10      D    Dflt  None     Dflt
tty11      D    Dflt  None     Dflt
tty12      D    Dflt  None     Dflt
```

Reporting login activity

To generate reports based on login activity, make the following **sysadmsh** selection:

Accounts ➪ Report ➪ Login

Login reports can be generated in three categories: by User, Group, and terminal.

Example 11-5 is a sample output listing login attempts by terminal.

Example 11-5 Sample login report by terminal

```
                       Login Activity Report
                           System  unix
                      Wed Mar 22 14:43:53 1991

          Last Good Login     Last Bad Login      Last Logout
Tty Name User Name Date     User Name Date     User Name Date     #Failed
--- ---- ---- ---- ----     ---- ---- ----     ---- ---- ----     -------
console  alvin     05/22/90 UNKNOWN   05/22/90 alvin     05/22/90 2
tty02    root      05/21/90 root      05/21/90 root      05/19/90 0
tty05    maryt     05/21/90 UNKNOWN   05/21/90 root      05/19/90 0
tty04    root      05/19/90 root      05/13/90 root      05/19/90 0
tty05    UNKNOWN   NEVER    root      05/13/90 UNKNOWN   NEVER    1
tty06    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty07    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty08    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty09    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty10    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty11    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
tty12    UNKNOWN   NEVER    UNKNOWN   NEVER    UNKNOWN   NEVER    0
```

Detecting system tampering

No system can be considered completely secure. When you consider that system penetration can be as simple as someone using an obvious password or leaving their terminal logged in overnight, you can see the user is the weakest link in the scheme. The system is designed to identify and authenticate users properly. In addition, access to security-related data on the system is based on subsystem authorizations. If a user has the proper authorization, then they can use system programs to modify the security databases (for example, the audit administrator changing the audit configuration, or the accounts administrator changing user passwords).

The system prevents unauthorized users from making such changes, but identification and authentication is a critical step in this protection. These mechanisms are circumvented when someone gains access to an account having greater authorization than their own. After having set up your system to minimize the possibility of tampering, the remaining task is to discover whether any tampering has taken place. Tampering can come from three avenues:

- Someone obtains a password of another user or gains access to their account (as when someone leaves their terminal logged in).
- A user with authorization abuses their privileges.
- A knowledgeable user gains unsupervised access to the computer system itself.

This section discusses how to detect such occurrences.

Stolen passwords

Each time a user logs in, the system displays the time and date of their last login. The most obvious evidence of a stolen password is when this last login is different from what the user remembers; secondary evidence is that a user's files have been altered. Warn users to note their last login and report any discrepancies immediately, including any instances where their files have been disturbed. Make certain that users follow the account usage guidelines discussed in the "Using a secure system" chapter of the *User's Guide*. These procedures ensure that other users cannot guess their passwords or otherwise obtain them.

The administrator should carefully consider what restrictions to place on passwords. One popular (and dangerous) practice is to have accounts without passwords. Although this feature is available, accounts without passwords are strongly discouraged. It is difficult to prevent damage or further penetration of the system once someone has logged on to an account. The identification and authentication procedure is the first line of defense against tampering.

Another weapon against stolen passwords is the interval between login attempts and the limits on unsuccessful login attempts for accounts and terminals. Although this can be annoying when a user makes a mistake in entering a password, it hinders a malicious user making repeated attempts to guess a password.

Other than reports from the users themselves, the principal method for detecting stolen passwords is to generate terminal and login reports as described in the "Activity report generation" section of the "Administering user accounts" chapter of this guide. Look for the following:

- logins made at off hours, or at times when the user is not supposed to be onsite
- accounts with multiple unsuccessful login attempts
- terminals with multiple unsuccessful login attempts

If any of these occur, you should suspect that someone is trying to gain access to your system. You should ensure that passwords are both changed regularly and made difficult to guess, which is the best assurance of password security.

Abuse of system privileges

If you have reason to believe a person with *root*-level authorizations is abusing their privileges, you should enable auditing for that user to determine if they are performing questionable actions. If the user has the *root* password or the **su** authorization, event L. (Admin/Operator Actions) will show the actions done that require high-level authorization.

Unsupervised access to the computer itself

The most basic security requirement is to prevent unsupervised access to the system itself. Although it is often necessary for users to access the console to operate the floppy or tape drives, it is dangerous to leave the computer hardware open to unsupervised access after hours. A knowledgeable user might be capable of disabling the system and penetrating the root account. This would be the most serious breach of security.

When no users are on the system, such an occurrence can go unnoticed. This kind of tampering can only be detected by looking for the following:

- login reports for the root account
- suspicious super user actions in the audit trail
- unexplained system reboots in the audit trail

The lesson here is to place your computer system under lock-and-key and disallow off-hour access to anyone other than designated system administrators.

Dealing with filesystem and database corruption

The cost of fixing a trusted system that has become untrusted is much greater than the cost of maintaining a trusted system. Once trusted, you can use a few procedures to monitor the integrity of the security perimeter. Filesystem corruption is an infrequent occurrence, but can result in the removal of files that are critical to the continued operation of your system. This notion of system integrity is different from dealing with tampering, which is the deliberate action of a malicious user to alter or access data. This section explains the important security database files and how to recover them in the event of a system crash.

The authentication database files

Several database files store the characteristics of the system itself, its users, its administrators, and its subsystems so that a site can control its own security parameters. These databases reside on the system and are maintained by an administrator. The format of these files is discussed in **authcap**(F).

WARNING The Authentication database files are not meant to be edited by hand. The trusted system utilities and **sysadmsh**(ADM) selections maintain and display the information contained in the databases. We do not recommend modification through any other means.

The Audit and File Control databases are independent databases. The other databases described here (the Protected Password database, the Terminal Control database, the Subsystem database, and the Device Assignment database) are referred to collectively as the Authentication database. The Authentication database is the responsibility of the authentication administrator, who has the **auth** authorization. Here are brief descriptions for each of the databases:

Audit
: controls the behavior of the audit system. This includes the types of activity, the system records on the audit trail, the performance/reliability attributes of the audit subsystem, and the filesystem devices on which audit information is collected. By changing parameters stored in the Audit database, the audit administrator can adjust the audit subsystem to suit the performance and security requirements of the site.

Device Assignment
: stores device pathnames relating to the same physical device. For example, */dev/tty1a* and */dev/tty1A* may refer to the same serial port with modem control disabled and enabled, respectively. This database is used by **init**(M) and **getty**(M) to stop one form of login spoofing, as described later.

Protected Password
: stores security information about each user. The user entry includes the encrypted password (which no longer appears in the regular password database */etc/passwd*) and password change, user authorization, and user audit parameters. By setting up this database properly, the authentication administrator controls how users identify and authenticate themselves to the system, the types of privilege users are allowed, and the extent to which users' actions are recorded in the audit trail. The System Defaults database, containing the system-wide default security parameters, is considered part of the Protected Password database.

Terminal Control
: gives access to the system through terminals. It records login activity through each attached terminal (last login and logout user, time stamps, and so forth). The Terminal Control database lets the authentication administrator set different policies on different terminals depending upon the site's physical and administrative needs.

Subsystem

is actually a series of files (one per subsystem) that store a list of users that are given special privilege either to be a subsystem administrator or to perform special functions within a protected subsystem. These files are another element of the Authentication database. It enhances accountability of administrative actions by allowing only specified users to run programs that maintain the internal subsystems. Security is enhanced by controlling who has permission to execute programs that maintain subsystems and by accounting for the real users that assume administrative roles.

File Control

helps maintain the integrity of the Trusted Computing Base. It does this by maintaining a record of the contents and protection attributes of files important to the TCB's operation. This database provides an effective tool for detecting modifications to the active copy of the TCB. The system administrator program **integrity**(ADM) checks the TCB file permissions against this database.

Checking the system after a crash

Several programs are used to maintain the Authentication database, the system area of the filesystem, and the filesystem as a whole. The basic rule is to work from the most basic components of the filesystem outward. Otherwise, corrections made at the higher levels may be undone by programs fixing the lower levels. Given this, use the programs in this section after a system crash or abnormality in this order:

1. Run a filesystem check with **fsck**(ADM) (automatic at reboot time).
2. Check for the absence of critical files with **tcbck**(ADM) (automatic at reboot time).
3. Generate an audit report (optional).
4. Check the consistency of the Authentication database with **authck**(ADM).
5. Check system file permissions with **integrity**(ADM).
6. Fix permissions with **fixmog**(ADM).

These programs should be run while the system is in single user (system maintenance) mode.

Using the override terminal

If your system is running with High security defaults, an override terminal entry exists for *root* in case the security databases become corrupted and all logins are disallowed. This is a special entry in the file */etc/default/login*. The entry identifies the tty to be used when doing an override login for *root*. The default entry (shown below) permits *root* to log in on */dev/tty01*, also known as the first multiscreen on the console. You can change this default to be another login device.

```
OVERRIDE=tty01
```

When the databases are compromised and *root* logs in on the override terminal, the following message is generated:

```
The security databases are corrupt.
However, login at terminal tty is allowed.
```

When the account is locked and *root* logs in on the override terminal, the following message is generated:

```
Account is disabled but console login is allowed.
```

The tty used should be physically secure; remember that normal locks do not apply to the super user account on this tty.

Filesystem checking: fsck(ADM)

Filesystems containing sensitive files must be considered sensitive entities themselves. Thus, the integrity of the filesystem afforded by the **fsck** program enhances the overall security of the system.

The **fsck** program must be run after any system crash or abnormal system termination. As always, make sure the system is in single-user mode when running **fsck**. There may have been user, system, or audit files in the process of being built when the system crashed. Although that data may be lost, **fsck** can recover some of those files in the *lost+found* directory of the filesystem, and at least fix basic filesystem problems.

fsck is discussed extensively in the "Managing filesystems" chapter of this guide.

Automatic database checking and recovery: tcbck(ADM)

When the system is halted suddenly by power or hardware failures, some filesystem damage can occur. Such damage can result in the removal of security database files, or can leave these files in an interim state if they were being updated at the time of the system crash. Whenever a reboot occurs, the system runs a series of programs to check the status of the database files. When the system terminates abnormally and is rebooted, this check is performed after **fsck**(ADM) is run on the root filesystem, prior to entering multiuser mode. This check is described in the "Checking the security databases" section of the "Starting and stopping the system" chapter of this guide.

NOTE **tcbck**(ADM) executes several checking utilities and even repairs certain inconsistencies (via **authck**), but does not execute **integrity**(ADM) or **fixmog**(ADM).

Database consistency checking: authck(ADM)

The **authck**(ADM) program checks the consistency of the Authentication database. (The functions of **authck** can also be accessed via **sysadmsh**, as discussed below.) There are several options that restrict the scope of the checking. See **authck**(ADM) for more information.

The Accounts menu of **sysadmsh** includes functions to check and repair inconsistencies in the Authentication database, a collection of files containing information about the trusted system.

Checking the authentication database files

To check the consistency of the password, terminal, and subsystem databases, make the following selection from **sysadmsh**:

Accounts ⇨ Check ⇨ Databases

This command uses the **authck**(ADM) command to check each database. When the **authck** command is executed from the command line, you are given the option of allowing **authck** to repair any inconsistencies. The Accounts ⇨ Check ⇨ Databases selection does not allow such repairs. If inconsistencies are found, you can execute **authck** from the command line or restore files if necessary.

Checking the */etc/passwd* and */etc/group* files

To check the consistency of the */etc/passwd* and */etc/group* files, make the following selection from **sysadmsh**:

Accounts ⇨ Check ⇨ Password

This selection does the same checking that is done when a new user is created. Error messages and warning messages are generated; any error messages must be acted upon.

System file integrity checking: integrity(ADM)

The **integrity**(ADM) program compares the entries of the File Control database against the actual file permissions on the system. It does not, however, alter permissions (see the section "System file permission repair: fixmog(ADM)").

You should run **integrity** as follows:

/tcb/bin/integrity -m -e > int.report

Print the file *int.report* and examine it. **integrity** reports files and directories that are missing or have incorrect permissions or ownership. Here are sample messages generated by **integrity**:

```
/etc/utmp (entry 83) is wrong.
        Owner is root, should be bin.
        Group is root, should be bin.
        Mode is 0644, should be 0664.
/usr/spool/lp (entry 233) is wrong.
        Group is bin, should be lp.
        Mode is 0755, should be 0070.
/etc/inittab (entry 71) is wrong.
        Type is d. should be r.
/usr/lib/mkuser/csh (wildcard entry 216) is wrong.
        Owner is bin, should be root.
        Mode is 0700, should be 0750.
```

The owner, group, and mode refer to the file permissions. The file types "d" and "r" refer to directory, and regular file, respectively. Missing files should be replaced by restoring them from backups. Permission and "type" problems can be fixed with the **fixmog** utility. All errors found during the integrity check are packaged as audit records that show the audit event as a Database Event in the audit trail.

NOTE Some files may be listed as missing in a correctly configured system, such as one of the pair */usr/lib/cron/at.allow* and */usr/lib/cron/at.deny*.

System file permission repair: fixmog(ADM)

The **fixmog** command attempts to correct inconsistencies found by **integrity**(ADM). **integrity** traverses the File Control database and compares each entry to the real file in the filesystem. Each file is checked to ensure it has the specified owner, group, access permissions and type. **fixmog** changes the owner, group and access permissions of files to the File Control database. You should always use the **-i** (interactive) option to ensure that you can confirm any changes before they are made.

Daemon operations on a trusted system

This section notes the features that affect system daemons, and lists examples of procedural and programming changes that must be made to add and run new daemons properly on a secure system. This ensures that the daemons are started with proper identity and privilege, encounter no surprises if the system acts differently due to security features, and handle boundary conditions and failure cases properly.

LUID enforcement

LUID enforcement requires that all processes have an LUID. Daemon processes that are **setuid** require special consideration on a trusted system. The only exceptions to the LUID rule are the processes that stamp the identifier on processes, namely the **init**(M), **login**(M), and **cron**(C) programs. (Technically, **getty**(M) also lacks an LUID, but it does not run set user ID programs). All trusted utilities either stamp their own LUID (for example, **auditd**(ADM)) or assume that their LUID was stamped before they run (for example, **lpsched**(ADM)). The **setuid**(S) and **setgid**(S) system calls fail if the LUID is not set.

The **cron** daemon is a special case and is allowed to run without an LUID. To start special daemons like **cron**, another daemon process, **sdd**, and a special utility, **sd**(ADM), are used to start and restart them. If you need to create a daemon that runs without an LUID, refer to the **sd**(ADM) manual page for more information.

NOTE If LUID enforcement has been disabled, use of the **sd**(ADM) command is unnecessary. See the "Disabling C2 features" section later in this chapter for details.

As administrator, you must ensure that every newly introduced daemon is stamped with an LUID if it is started from the system startup files (*/etc/rc?.d/**). The proper procedure is to set up the */etc/passwd* and */etc/group* files with the proper pseudo-user and group accounts, and the Protected Password entry for the account. If the daemon is to be run from a startup script, add a line to that script, as shown below, that runs the program from **su**(C) so that the identity of the process is set properly. The procedure is the same as running daemons under a certain account using the traditional startup scripts. For example, the line printer daemon **lpsched** is started with the following line:

```
su lp -c /usr/lib/lpsched >/dev/null 2>&1
```

The trusted version of **su** program sets the LUID for a process if it has not already been set.

stopio(S) on devices

Note that the standard output and error of the sample **lpsched** command was redirected to the null device. The system has a feature that makes it difficult to handle console output from a daemon, and you must plan daemon output accordingly. All terminal devices are subject to the trusted system call, **stopio**(S), which was added to enhance the identification and authentication subsystem to prevent login spoofing. When a user logs out, the **getty** that is respawned on that terminal line calls **stopio** with the terminal device name as argument. Any processes holding that device open are killed (signal **SIGHUP**) if they try to write to the device again. Daemons that write to the console are subject to this signal if a logout occurs at the console between daemon start up and daemon output. Because most daemons ignore **SIGHUP**, their message output is simply lost. Therefore, you should redirect daemon output to a file or disabled terminal if it must be preserved, or redirect the output to the null device as in the above example.

NOTE The use of **stopio**(S) on devices can be disabled if desired. See the "Disabling C2 features" section later in this chapter for details.

Processes in the operating system run with a set of kernel authorizations that control the special rights the process has for certain privileged system actions. If the daemon must take an action that requires one of those privileges, that account must be set up properly so that those privileges are applied to the daemon process. Kernel authorizations are discussed in "Authorizations" under "Trusted system concepts" earlier in this chapter. If a daemon executes other SUID programs, it must have the *execsuid* authorization. If the process creates files with the SUID bit, it must have the *chmodsugid* authorization. If it uses **chown** to alter ownership of files, it must have the *chown* authorization. No processes that are not installed with the TCB should run with any of the audit authorizations. Other authorizations are for special situations, and should not be allowed to non-TCB daemons.

Sticky directories

The final feature that may affect daemons is sticky directories. If a directory's mode includes the save text (sticky) bit, only the owner of the file or root can remove the file from the directory. Daemons that manipulate temporary directories may behave improperly if files that they had assumed they could delete cannot be deleted.

You can handle this situation in one of two ways. First, remove the directory's sticky bit. This solves the daemon problem, but users must be cautioned of the security implications of using that directory for holding temporary files. The other solution is to modify the daemon and its corresponding helper program to agree on a new convention for file sharing. This second situation assumes that you have source code available and that you have the expertise and budget to modify the application.

You must carefully consider each daemon program so that it can run with proper behavior and safety. You should carefully test the daemon in a controlled environment and observe that it acts properly before opening it up for general use. This leads to fewer security problems introduced into your system, and fewer surprises when users attempt to use the daemon and receive unexpected results.

Disabling C2 features

In addition to customizing security parameters, certain key C2 features can be disabled completely (in the Low and Traditional security defaults they are disabled by default). This functionality is provided to aid in resolving incompatibilities that expect traditional UNIX system behavior. The following key features can be "turned off" by changing an associated kernel parameter:

LUID enforcement

Under C2 requirements, every process must have a login userID (LUID). This means that processes which set UIDs or GIDs, such as the printer scheduler (**lpsched**), must have an LUID set when started at system startup in */etc/rc2.d*. This can cause problems with **setuid** programs. When the security mode is set to a lesser mode (that is, not "High"), enforcement of login user ID (LUID) is relaxed and **setuid** programs do not require an LUID to run. This feature is enabled by default when the High security default is selected, but it can be enabled or disabled by modifying the **SECLUID** kernel parameter. A value of 0 disables the enforcement of LUID.

Clearing of SUID/SGID bits on write

Under C2 requirements, the set user ID (SUID or **setuid**) and set group ID (SUID or **setgid**) bits on files must be cleared (removed) when a file is written. This prevents someone from replacing the contents of a **setuid** binary. This can cause problems with programs that do not expect this behavior. In the lower security defaults, SUID and SGID bits are not cleared when files are written. This feature is enabled by default when the High security default is selected, but it can be enabled or disabled by modifying the **SECCLEARID** kernel parameter. A value of 0 disables this feature.

stopio(S) on devices

The **stopio**(S) call is used under C2 to ensure that a device is not held open by another process after it is reallocated. This means that other processes attempting to access the same device are killed. This can cause problems, for example, when two **lp** processes attempt to access */dev/null*. In the lower security defaults, **stopio**(S) is not called. This feature is enabled by default when the High security default is selected, but it can be enabled or disabled by modifying the **SECSTOPIO** kernel parameter. A value of 0 disables this feature.

These parameters can be changed by invoking the **sysadmsh** selection System ⇨ Configure ⇨ Kernel ⇨ Parameters and selecting category 14: "Security," and changing the parameter desired. The kernel must then be relinked and booted for the new behavior to take effect. Use the **sysadmsh** System ⇨ Configure ⇨ Kernel ⇨ Rebuild selection to relink the kernel. See the "Reallocating kernel resources with configure" section of the "Tuning system performance" chapter of this guide for complete instructions.

Chapter 12
Using the audit subsystem

The audit subsystem records security-related events that occur on a system in the form of an "audit trail" that can later be examined. Audit trails produced by this subsystem can detect penetration of the system and the misuse of resources. The audit subsystem is designed to meet the audit goals specified by the U.S. National Computer Security Center.

Auditing permits the review of the collected data to examine patterns of access to *objects* (files) and to observe the actions of specific users and their processes. Attempts to violate protection and authorization mechanisms are audited. The audit subsystem provides a high degree of assurance that attempts to bypass security mechanisms are audited. Because security-related events are audited and are accountable to a specific user, the audit subsystem serves as a deterrent to users attempting to misuse the system.

The audit subsystem uses system call and utility usage to classify user actions into event types. These can be used for selective audit generation and reduction. One such event type, *Discretionary Access Control* (DAC) Denial, records attempts to use objects in a manner not permitted by the object's permissions. For example, if a user process attempts to write a read-only file, a DAC Denial event is audited, showing that the process tried to write a file to which it was not entitled. When you examine the audit trail, it is easy to notice repeated attempts to access files for which permission is not granted. This alerts the administrator to possible tampering or penetration.

Essential to the effectiveness of the audit data is the ability to uniquely identify all users and their actions so that the audit trail accurately reflects the auditable actions of the each user. As users attempt to log onto the system, they must go through an identification and authentication process before access to the system is granted. The security mechanism stamps each process created by a user with an immutable indicator of identity: the login UID or LUID. The LUID is preserved regardless of transitions between user accounts with commands like **su**(C). Each audit record generated by the subsystem contains the LUID together with the process's effective and real user and group IDs. As a result, users can be held strictly accountable for their actions.

The audit subsystem is administered by the audit administrator. The audit administrator has complete control over the events selected for audit record generation, over the parameter values for subsystem control, and over the subsequent reduction and analysis of audit data.

This chapter explains the following:

- introduction: understand the principles of auditing and the design of the audit subsystem.
- data collection: select audit criteria, enable and disable auditing, and adjust audit performance parameters.
- file and directory management: set storage of audit records, back up and remove audit records, monitor disk usage.
- report generation: use audit report templates, generate audit reports, and interpret the data.

An audit glossary is included at the end of this chapter to explain the terms used.

Audit subsystem components

The Audit Subsystem consists of five major components:

- kernel audit mechanism
- audit device driver (*/dev/auditr* and */dev/auditw*)
- audit compaction daemon (**auditd**(ADM))
- **sysadmsh** audit interface
- data reduction and analysis facility

Although not actually part of the audit subsystem proper, there are also a number of trusted system utilities responsible for writing audit records to the audit trail (such as **login**(M)).

Kernel audit mechanism

The kernel audit mechanism is central to the audit subsystem. This mechanism generates audit records based on user process activity through kernel system calls. Each kernel system call contains an entry in a subsystem table that indicates whether the call is security-relevant and, if so, to what event type the system call corresponds. Additionally, a table of error codes further classifies the system calls into specific security events. The kernel audit mechanism makes an internal call to the device driver to write a record to the audit trail.

For instance, the **open**(S) system call is classified as a *Make object available* event. If user *blf* performs an **open**(S) on */unix* and it succeeds, an audit record is generated indicating that event. However, if the system call fails because *blf* requested write access on the **open**(S) but does not have write permission on the file, the action is classified as a DAC Denial event for *blf* with object */unix*. Consequently, a system call can map to a number of event types, depending on the object accessed and/or the result of the call. It is, therefore, possible that a system call might be audited selectively, depending on the event types that you enable.

Some system calls are not considered relevant to security. For instance, **getpid**(S) retrieves the process ID of a process and does not constitute an event of security relevance. Thus, that system call is never audited.

Audit device driver

The audit device driver is responsible for the following:

- accepting audit records from the kernel audit mechanism and from trusted utilities
- creating and writing the intermediate audit trail files
- providing audit trail data to the audit daemon for compaction
- providing for selective audit record generation based on event types, user IDs, and group IDs

The device driver provides **open**(S), **close**(S), **read**(S), **write**(S), and **ioctl**(S) interfaces like many other character devices. (The audit device is described on the **audit**(HW) manual page). However, the audit device can only be opened by processes having **configaudit** or **writeaudit** kernel authorizations. This limits access to the audit device only to trusted utilities such as the audit daemon and the audit administrator interfaces. The audit device can be written to by many processes at the same time. The device handles the merging of the records into the audit trail. The device can only be read by a single process, the audit daemon.

The audit device driver maintains the audit trail as a set of *audit collection files.* Each time you enable auditing, a new audit session is begun. As the session starts, the subsystem creates a collection file into which audit records are written. When the collection file reaches a certain size (configurable by the administrator), the subsystem creates a new collection file and begins writing to it. The audit trail could, therefore, be viewed as a continuously growing sequential file even though many collection files are used. That is precisely how the audit trail is viewed by the audit daemon, because it reads the device and is presented with records from the audit trail. The subsystem handles the necessary switching to new collection files for the daemon when the end of a file is reached. All of this is transparent to the daemon.

Audit compaction daemon

The audit daemon **auditd**(ADM) is a trusted utility that runs as a background daemon process whenever you enable auditing. The daemon is the sole reader of the audit device which, in turn, provides the daemon with blocks of records from the audit collection files. The daemon is not concerned that the audit trail is spread over numerous collection files. The audit device driver satisfies the read requests from the daemon and handles the switching and deletion of collection files as needed.

The main purpose of the daemon is to provide a compaction and logging mechanism for the audit session. The daemon also serves a support role for protected subsystems, enabling them to write audit records to the subsystem. Depending on the audit record generation criteria you select, a large amount of audit data can be generated on the system. For a typical single-user system, it would not be uncommon to generate 200 Kbytes of audit data in an hour. The daemon, therefore, provides a compaction mechanism, compressing the audit data into a packed record format that is stored in an *audit compaction file.* The compaction algorithm provides for an average 60% reduction in file space. This greatly reduces disk space used to store audit records.

A second function of the daemon is to provide a log file describing the current audit session. The log file contains information about the number of audit records available in the compacted file's output for the session; the start and stop times of the session; and other indicators pertaining to the audit session's state. Just as the audit device driver switches collection files as they reach specified sizes, the daemon can create multiple compaction files to avoid growing a single file too large to be manageable. (This is also configurable.) Audit compaction files written by the daemon may also be located in a variety of administrator-specified directories. For these reasons, the log file is maintained to provide a trail of compaction files that can be used for subsequent data reduction.

A third function of the audit daemon is to serve as an interface program to the audit device driver for the writing of audit records from protected subsystems that do not have the **writeaudit** authorization. Because these subsystems cannot access the audit device driver directly but can interface to the daemon in a trusted manner, the daemon handles the writing of the application audit record to the subsystem.

Audit subsystem interface

sysadmsh presents simple options to set up and maintain the audit subsystem. This allows the administrator to handle setup and initialization, modify subsystem parameters, maintain the subsystem (backup, restore, and so on), and reduce both general and selective audit data.

Data reduction and analysis facility

The audit subsystem also includes a data reduction and analysis facility to examine audit trails from previous audit sessions or from the current audit session. By using the log file produced by the audit daemon, the reduction utility can identify all of the compaction files needed to reduce an audit session. Because the compaction files are in a compressed format, the reduction program contains the necessary routines to uncompress the data.

To provide effective analysis of audit data, the reduction utility lets you specify certain event types, user IDs, group IDs, and object names to reduce the data selectively. In addition, you can specify a time interval to be applied while searching for records to match the specified criteria. If a record is not within the specified time interval, it is discarded for the purpose of that reduction.

As an example, you may reduce the data selecting the DAC Denial event with user ID *blf* looking for the object */unix*. Only records that reflect an access attempt to */unix* by *blf* that was denied because of lack of permission are printed. This provides a powerful mechanism for identifying security events of immediate interest without having to analyze the entire audit trail.

Audit methodology

This section explains how the audit subsystem functions, what criteria are used to collect data, and how audit requirements affect system performance.

Audit authorizations

There are four authorizations associated with the audit subsystem:

- The **configaudit** kernel authorization allows the audit parameters for all users of the system to be set.
- The **writeaudit** kernel authorization allows specific information to be recorded in the audit trail.
- The **suspendaudit** kernel authorization prevents any auditing.
- The **audittrail** secondary subsystem authorization allows users to generate audit reports on their own activities. When a user is assigned this authorization, they can access the System ➪ Audit ➪ Report functions.

Audit record sources

The audit trail contains the security-related events for the system. Effective auditing concerns not only system call requests from user processes but also certain events such as login, logoff, and login failure attempts. These events are critical to determining who has accessed the system, at what times, from what terminal, and what actions were performed. Login failures are impossible to audit at the kernel level because the kernel has no knowledge of what an application is specifically doing. Thus, certain security-critical utilities such as **login** must be allowed to generate audit records.

Audit records are generated from three sources (discussed in the sections that follow):

- kernel audit mechanism
- trusted application processes
- authorized subsystems

Kernel audit mechanism

A large percentage of the audit records stored in the audit trail are generated by the kernel audit mechanism. This portion of the audit subsystem generates records in response to user process system calls that map to security-related events. Some system calls, **open**(S) for instance, map to multiple security events depending upon user arguments and the state of the file being opened. If **open**(S) is called with the **O_CREAT** flag, the file is created if it does not exist. If the **O_TRUNC** flag is specified, the file is truncated to zero length if it exists. This illustrates how the **open**(S) call could map to one of three distinct events, *Make Object Available*, *Object Creation*, or *Object Modification*.

Error codes also play an important role in determining the event. Errors on system calls that indicate access or permission denials as well as resource consumption problems are mapped to specific event types. The kernel audit mechanism determines at the end of the system call what event class the call belongs to, and if that event is to be audited as specified by you. In addition, the mechanism may apply additional selection criteria such as user ID or group ID. In this manner, the generation of audit records can be limited to a select group of users.

Trusted applications

The Trusted Computing Base (TCB) contains a number of trusted applications essential to providing a trusted environment. Among these are **login**, **su**, and various audit subsystem commands. To reduce the amount of audit data written to the audit trail, and to make the trail more meaningful, these trusted applications are permitted to write directly to the audit device. This enables **login**, for instance, to write a login audit record to the audit trail rather than letting a login on the system be represented as a collection of system calls required to complete the login procedure.

It is not sufficient to just let the trusted applications write to the audit device. There must also be a way to suppress the generation of system call audit records by the kernel audit mechanism to avoid the problem of a cluttered audit trail. Thus, the **suspendaudit** authorization exists as discussed earlier. Trusted applications run with this authorization enabled, suspending kernel system call auditing for that process and allowing it to open and write the audit device. Only a few trusted applications are permitted to do this. A user process should not run with **suspendaudit** authorization. The authorization mechanism is managed by **login**, using restricted system calls, and is based on Protected Password database entries.

Authorized subsystems

A third method in which audit records are generated is through authorized subsystems such as the *lp*, *cron*, *terminal*, and *mem* subsystems. Sometimes, a subsystem encounters inconsistencies or problems that make the writing of an informative audit record desirable. However, subsystems do not possess the **writeaudit** authorization and cannot directly write audit records to the subsystem.

Instead, the subsystems format the records just as a trusted application would, and present the records to the audit daemon process through a trusted interface. The audit daemon, which is a trusted application, performs the task of writing the audit record to the audit device. This allows concise and informative audit records to be generated by protected subsystem processes without having to distribute the **writeaudit** authorization to these systems.

Accountability for audit

The audit subsystem audits security-related system events and associates the events with a specific user. Users log into the system through the **login** program. This program performs authentication on the user to determine whether access is permitted. The login procedure has been enhanced to provide audit support for both successful and unsuccessful login attempts. When a user is successfully logged in, **login** stamps the user process with the login user ID (LUID). Regardless of the number of **setuid**(S) and **setgid**(S) system calls made by that process, the LUID does not change. Strict accountability is maintained for the process and the user. A user process may still perform **setuid** and **setgid** system calls, which are also audited. The audit records indicate the LUID of the process together with the effective and real user and group IDs of the process.

Audit event types

Every audit record, regardless of the originator, is stamped with an event type. For user process system calls, the event type is determined by the kernel audit mechanism, based on the system call and its outcome as previously discussed. For application or subsystem auditing, the process writing the audit record sets the event type. This event type is not changed by the audit device or by the audit daemon.

Event types are important because they classify the security event on the system. Both audit-record generation and reduction can be controlled based on event types. For example, if you are only concerned with users logging onto and off the system, you can specify that event type for collection or reduction.

The audit subsystem provides a wide range of event types that strike a balance between granularity and relevant security classes. These events are summarized in Table 12.1; the letters are a simple identifier used to refer to the event by the audit subsystem.

Table 12-1 Audit events

A.	Startup/Shutdown	B.	Login/Logoff
C.	Process Create/Delete	D.	Make Object Available
E.	Map Object to Subject	F.	Object Modification
G.	Make Object Unavailable	H.	Object Creation
I.	Object Deletion	J.	DAC Changes
K.	DAC Denials	L.	Admin/Operator Actions
M.	Insufficient Authorization	N.	Resource Denials
O.	IPC Functions	P.	Process Modifications
Q.	Audit Subsystem Events	R.	Database Events
S.	Subsystem Events	T.	Use of Authorization

You can selectively collect and reduce audit data based on these event types. The audit subsystem interface lets you build a list of event types for either the audit subsystem or the data-reduction program.

The subsystem uses event types to determine whether an audit record should be written to the audit trail. As the audit administrator, you have full control over what events get audited.

To control event type auditing, the subsystem contains a global *system audit event mask,* as explained below. The audit subsystem also maintains a mask of event types for each process on the system (explained in a later section).

Mandatory auditing

To maintain accurately all the required information about a user process for meaningful audit output, the kernel audit mechanism always audits certain system calls. When auditing is enabled, this means that some events are audited even if no events were selected by the audit administrator. These are known as mandatory system calls. They are essential to the maintenance of the process state. For example, the **open**(S) system call may specify a relative pathname such as *../newfile*. The full pathname depends on the current directory of the process, which is set using the **chdir**(S) system call. The audit record containing the pathname *../newfile* could not be meaningfully reduced without prior knowledge of the value of the current directory.

The problem applies to the **close**(S) system call as well. This system call requires only a file descriptor as the argument to close a previously opened file. The **close** audit record would be insignificant unless the name of the object being closed is output in the record. However, unless the pathname is retained when the file is opened, there is no way to provide the pathname for the close. Table 12.2 lists the audit event types affected.

Table 12-2 Mandatory audit events

Event type	Always audited	Optionally audited
Make object available	open, pipe	mount, opensem
Object creation	creat	link, mkdir, mknod, creatsem, sdget
Map object to subject	dup, exec, exece	fstatfs, getdents, stat, statfs
Object modification	execseg, unexecseg	chsize, stime
Make object unavailable	close	sdfree, umount
Process create/delete	exit, fork	-
Process modification	chdir, chroot, proctl, security, setgid, setpgrp, setuid	-

Mandatory auditing is not limited to just the group of system calls listed in Table 12.2. The login event is the only mandatory trusted application audit record defined. When a user logs in, the login record contains an indicator of the terminal on which the login occurs. If that same user is logged into multiple terminals on the system, the actions of that user can be traced to a specific terminal.

System audit event mask

The system event mask is global to the audit subsystem. You can change it during auditing if you want to select a different set of events. The system event mask contains one bit for each event type; the bit is set to one when auditing is desired. This provides a fast test (using a bit-wise operation) to determine if a newly created record is enabled for auditing. The audit subsystem uses the system event mask to compute user masks when a new process is created through a login.

User-specific and process event masks

You can override the system-wide event mask for any user by setting up a *user-specific event mask.* Each process on the system has a *process event mask* that tells the system what to audit for that process. When a user logs in, the **login** program looks up the user-specific event mask and sets the process event mask for the login shell as discussed here.

The user-specific event mask has one of three values for each audit event type:

- Always audit this event.
- Never audit this event.
- Use the system audit event mask.

For each audit event type, the process audit mask is set from the user-specific mask if it indicates that the event is always or never audited. Otherwise, the process audit mask is set from the system audit event mask. In most cases, the user-specific event mask is set to the third value for all audit events, which causes the system default to apply to that user. You can use the user-specific mask to audit either more or less information about users that you trust either more or less than the rest of the user population.

Guidelines for effective system auditing

You should follow certain guidelines to use the audit subsystem. The subsystem is designed to offer flexible performance and reliability and to let you collect the audit data that you want. Audit-record generation supports *preselection* of audit events, user IDs, and group IDs. Preselection is valuable if you want to concentrate on a specific user or group of users for some reason (when particular users have a pattern of attempting access to files to which they are not permitted). Event types may also be used for preselection such as auditing only login and logoff events. Preselection also provides disk-space savings benefits because the amount of audit records written to the collection files by the subsystem is reduced. There is, however, a drawback to using preselection. If a system security violation occurred and that event or the user that perpetrated the event was not selected for audit, the record of the action is lost.

For this reason, it is more conservative not to preselect the audit events and users or groups, but instead to perform full auditing. The benefit is that any security-related event that occurs is recorded in the audit trail. The disadvantage of full auditing is that it consumes a great deal of disk space.

You can then combine full auditing with *postselection* to examine only records of interest. Post-selection provides for the selective examination of the audit trail based on event types, user IDs, group IDs, and object names, as well as date and time of record generation. In all, the audit subsystem combined with the data reduction/analysis utility provides you with the flexibility to trade between system performance and disk capacity with preselection, and the convenience of full auditing combined with post-selection.

The administration of the audit subsystem is the key to effective auditing. Through careful setup and use of the audit subsystem, you have a powerful tool that helps keep the system trusted and identify problems when they do arise. The subsystem is designed to be very complete in terms of audit event coverage both from kernel actions and from the use of system utilities. It is also designed for reliability and to minimize the impact on the performance of the system as a whole.

How well the subsystem meets your goals depends on proper administration of the system. You control the tradeoff between reliability and performance using audit parameters. Improper setup can result in poor performance, loss of audit data, or both. For example, setting the audit event mask to govern event types audited by the subsystem is critical. For instance, if event preselection does not include login events, a penetration of the system through a dial-up line might go undetected. Therefore, it is vital that you carefully consider the following three items:

- performance goals
- reliability goals
- security goals

Performance goals

When estimating the impact of the audit subsystem on the performance of the system, it is important to consider the actions that must be performed by the subsystem. The audit subsystem device driver is the focal point for the collection of audit records from all sources and is responsible for writing those records to the audit trail. The driver writes to a collection file that is shared by all processes being audited in the system. This situation is similar to an airline reservation system where multiple clerks are accessing a common database. Lockout mechanisms must exist to prevent the intermixing of audit records and to insure the consistency of the database. The same is true of the audit subsystem collection files.

An internal buffering mechanism and a write-behind strategy tries to minimize the impact of multiple, simultaneous writers to the collection file. This lets the subsystem service audit records from processes and applications while collection files are being written in parallel. You can tune this mechanism for how much buffering is used and how frequently data is written to the collection file.

Reliability goals

Equally important to the system's performance is the reliability of the audit trail produced. Traditional UNIX systems lack the element of preserving filesystem integrity when a system crash occurs. This stems from the fact that I/O is accomplished using a pool of buffers that are (mostly) written asynchronously. Thus, changes made to files may not actually be recorded on disk at the time of a system crash.

This is unfortunate because the events leading up to a system crash are the ones that are most interesting from an audit standpoint. It is highly desirable to minimize any potential data loss from the audit subsystem as the result of a system crash. To meet this objective, the audit subsystem uses a facility called synchronous I/O that causes audit collection buffers and collection file inodes to be updated immediately as they change. This minimizes the potential amount of data that could be lost as the result of a system crash.

There is a direct correlation between the degree of data reliability and the performance of the audit subsystem. Audit records that are generated by the kernel audit mechanism, trusted applications, and protected subsystems are typically 40 to 60 bytes in length. If each record is written to the disk synchronously as it is presented to the subsystem, the result is poor performance; the I/O system gets flooded because of the high rate at which these records are generated. The solution is to buffer the records and write them together to the audit trail at selective intervals. These intervals can be determined by elapsed time or an accumulated data threshold. Again, the choice is yours.

Security goals

The final area critical to audit subsystem administration is determining what needs to be audited. Preselection options for record generation can be used to fine tune the audit trail to concentrate on an event or several events. For instance, the system may be limited in use to a small group of people but left unattended at night. Additionally, several dial-in lines may be provided for after-hours work. You may only be concerned with accounting for who uses the system and when. In this case, preselection can be used only to audit login and logoff events. Attempts to penetrate the system by unauthorized users would then be audited as unsuccessful login attempts.

Audit may also be focused on specific users or groups of users. This lets you concentrate on suspected violators of security policies. The less auditing that is requested, the less impact the subsystem has on the system performance.

Full auditing creates an extensive and detailed record of system events, but also requires the most resources to accomplish. However, it is often better to have recorded the events and to use the reduction tools to discard unwanted records later than not to have the records that are really needed to examine a problem. This decision depends on the degree of security you wish to achieve.

It is important to understand the definition of an audit session with respect to the subsystem. A session is intended to correspond to an interval from the time the system is booted until the system is taken down. To reduce the amount of data written to the audit trail, the audit subsystem was designed to minimize the size of each audit record. Consequently, the state of a process is defined by a sequence of audit records rather than being indicated completely in each record. The space and time savings of this approach are tremendous but require that careful administration be used to avoid pitfalls.

WARNING If the audit subsystem is disabled while the system is running and later re-enabled, a new session is created. A session is defined as the sequence of collection and compaction files containing the audit records associated with a specific time interval. Some processes that are audited in the second or subsequent session might have been created during the first session. Consequently, a session may not contain all of the relevant process state needed for a certain process. In turn, this can lead to incomplete record reduction. This applies mostly to filenames and typically only in the case of relative (rather than absolute) filenames. You can avoid this problem by disabling auditing only by taking the system down. For more information, see "Maintaining audit trail continuity" later in this chapter.

Administrative concerns

This section discusses particular areas of concern for the audit administrator.

Disk space

The audit subsystem can generate a large number of audit records. Even though the records are fairly small, the storage required to maintain them can grow quite large. As a consequence, care must be exercised in administering the system. Auditing should be directed to disks that have a good deal of space available. The subsystem has built-in protection mechanisms that warn when the audit device is getting low on space. If the situation is not rectified and the amount of disk space remaining goes below a certain threshold, the subsystem attempts to switch to a new audit directory. For this reason, alternate audit directories should be placed on different filesystems. Whenever the subsystem encounters an I/O error, it attempts to audit to a new directory in the list.

System failures

Most systems crash at some time, despite every effort to provide a resilient base. If a system crash occurs, there is potential for data loss in the audit trail due to buffered output records and inode inconsistencies. The audit subsystem makes every attempt to use synchronous I/O for critical operations like buffer, inode, and directory flushing. However, this does not guarantee that data always makes it to the disk. This is especially true if a disk failure causes the system crash.

It is not uncommon to find filesystem damage on audit trail files upon re-boot. You may have no choice but to remove the audit files to clear up the problem. This compromises the audit trail somewhat, but should pose no problem for recovering the filesystem from whatever damage occurred.

Subsystem messages

The audit subsystem is resilient. I/O errors are handled by the subsystem by attempting to switch collection or compaction to a new directory. The same is true of recovery in cases where filesystem free space gets too low. There are situations where the subsystem may be unable to continue. If the disk media is corrupted or there is no filesystem space remaining, the subsystem terminates and prints a message to that effect on the system console. Any abnormal termination condition results in a console message that should help you determine the problem.

In the case of system problems in general, the symptoms are not generally limited to audit alone. One problem that can occur upon removal and subsequent re-creation of the audit parameter file relates to duplicate session-building. Each time auditing is enabled, a new session is created. The session is defined by the log file and all of the compacted files generated during the audit period. The files are uniquely stamped with the session number for easy identification and use by subsystem utilities that need access to the files; the utilities may deal with session numbers rather than filenames.

If sessions are allowed to remain on the machine and the parameter file is modified such that the subsystem session number is reset, the result may be an attempt to create an audit file using the same name as a previous session. If this occurs, the old sessions should be archived and removed using the System ⇨ Audit ⇨ Files functions of **sysadmsh** before auditing is re-enabled.

Auditing as a debugging tool

Another useful aspect of auditing is in debugging programs. Because an audit session can log specific activities, you can enable auditing while running a troublesome program and find out exactly what it was doing.

Data collection

This section explains how to set up, activate, manage, and deactivate auditing. The system is distributed with default collection parameters (see "Default account configuration" in the "Administering user accounts" chapter of this guide for a list of defaults). You can modify these defaults to fit your needs.

As discussed in the "Introduction," usage of the audit subsystem has two stages, collection and reporting. Each stage involves selection of audit data. Data collected by the audit subsystem is governed by a set of parameter files called masks. The word "mask" is used because unwanted data is "masked out", and not collected. Once initiated, the subsystem collects data as directed by the audit masks until auditing is terminated, or the system is halted. The system maintains two types of masks: the system-wide mask that governs default auditing done on all users, and an individual user mask that can be defined for each user. The user mask overrides the system-wide mask.

The following data collection functions are available:

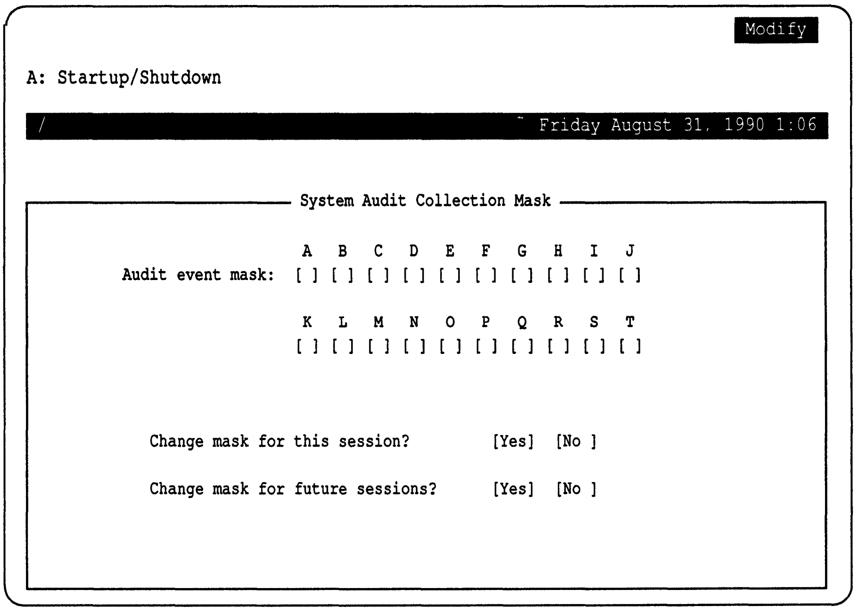
Directories	display or modify audit collection and compaction file directory list.
Events	display or modify system audit collection type masks.
IDs	display or modify list of users and groups audited.
Parameters	audit subsystem performance parameters.
Reset	change collection rules back to the default values.
Statistics	display statistics of current audit session.

Choosing audit events

To select events to audit on a system-wide basis, make the following **sysadmsh** selection:

System ➪ Audit ➪ Collection ➪ Events ➪ Modify

You see a screen similar to the following:



```
                                                          Modify

A: Startup/Shutdown

 /                                      Friday August 31, 1990 1:06

  ------------------ System Audit Collection Mask -----------------

                             A   B   C   D   E   F   G   H   I   J
        Audit event mask:   [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]

                             K   L   M   N   O   P   Q   R   S   T
                            [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]


          Change mask for this session?       [Yes]  [No ]

          Change mask for future sessions?    [Yes]  [No ]
```

Each event type displayed corresponds to a letter. For those events that are to be audited, the event type should be specified with a "Y". Those event types that are not to be audited are excluded using the "N" option. Use ⟨Space⟩ to toggle an entry from "Y" to "N" and vice versa. Use the arrow keys to move from entry to entry. This event mask can be modified and dynamically altered for the current audit session and/or can be written to the parameter file to take effect on future audit sessions.

As discussed previously under "Audit event types," there are a number of audit events that can be selected; these are summarized in Table 12.3.

Table 12-3 Audit event types

Event Type		Description
A.	Startup/Shutdown	system startups (boots) and shutdowns
B.	Login/Logoff	successful and unsuccessful login attempts
C.	Process Create/Delete	creation and termination of processes
D.	Make Object Available	file, message, semaphore opens and filesystem mounts
E.	Map Object to Subject	program execution
F.	Object Modification	file writes
G.	Make Object Unavailable	file, message, semaphore closes and filesystem unmounts
H.	Object Creation	file/message/semaphore creation
I.	Object Deletion	file/message/semaphore deletion
J.	DAC Changes	file, message, semaphore permission or ownership changes
K.	DAC Denials	denied permissions
L.	Admin/Operator Actions	system administrator and operator tasks
M.	Insufficient Authorization	tasks that failed due to insufficient privileges
N.	Resource Denials	missing files and insufficient memory
O.	IPC Functions	sending signals and messages to processes
P.	Process Modifications	effective identity or working directory changes
Q.	Audit Subsystem Events	system auditing enable, disable, modification
R.	Database Events	security data changes and integrity
S.	Subsystem Events	use of protected subsystems
T.	Use of Authorization	super user-only actions

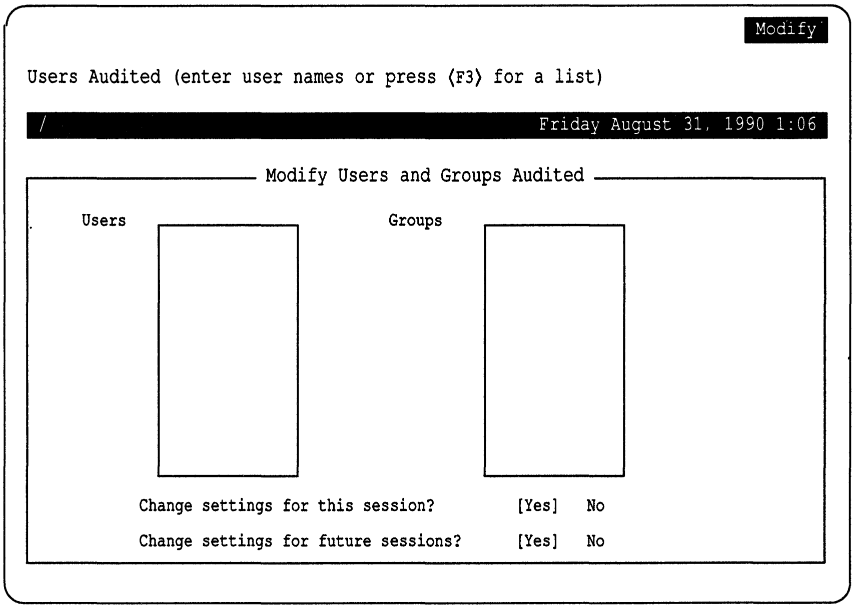
Auditing individual users and groups

The User and Group fields can dynamically alter the audit selection for the current session or can affect the next session. Selection of users and groups can be done many times within the same session. If no users and groups are selected, all processes are subject to the system-wide audit mask.

To audit by users or groups, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Collection ⇨ IDs ⇨ Modify

A screen similar to the following is displayed:



```
                                                              Modify
Users Audited (enter user names or press <F3> for a list)

/                                              Friday August 31, 1990 1:06

              Modify Users and Groups Audited

   Users                           Groups

              Change settings for this session?       [Yes]   No

              Change settings for future sessions?    [Yes]   No
```

Displaying current audit statistics

An option is provided for the retrieval of the current audit session statistics, providing information on the current session number, the number of collection and compaction files, the number of records written by the kernel audit mechanism and the number written by applications, as well as other information. If auditing is not currently in effect, no statistics are displayed.

To display the statistics of the current audit session, make the following **sysadmsh** selection:

System ➪ Audit ➪ Collection ➪ Statistics

Example 12-1 Audit collection summary example

```
            *** Audit Subsystem Statistics ***

    Current Audit Session-6
    Current Collection File Sequence Number-1488

    Total count of audit data written:        7659433
    Total count of audit records written:     156666
    Audit records written by applications:    81
    Audit records written by system calls:    155083
    System calls not selected for audit:      751889
    Total number of audit device reads:       2977
    Total number of audit device writes:      324
    Total number of collection files:         1489
    Highest number of audit buffers used:     1
    Total number of audit buffer sleeps:      0
```

Enabling and disabling auditing

To switch auditing on or off, use the following **sysadmsh** selections:

System ➪ Audit ➪ Enable
System ➪ Audit ➪ Disable

The enable function uses the current audit parameter file to perform the subsystem initialization. The disable function is available from the same menu and causes a graceful exit from auditing (at which point all collection files have been read by the daemon and compacted). The daemon then terminates leaving only an audit session log file and the session compaction files.

Most subsystem parameters can be modified while auditing is running, so you do not need to disable audit for that purpose. Both enable and disable functions have confirmation screens that must be acknowledged before the function is completed by **sysadmsh**. When auditing is enabled or disabled, a message is displayed indicating the status of auditing at reboot time; if disabled, auditing will be disabled at system startup and if enabled, auditing will be enabled again at startup.

Maintaining audit trail continuity

There is an important consideration involving LUIDs and audit sessions. Example 12-2 is a section of an audit report that shows a denied file access, but with a user ID of *root* and not the unauthorized user ID that actually tried to access the file (in this case to **touch** the file */a*).

Example 12-2 Incomplete audit trail example

```
Process ID: 227 (*INC*) Date/Time: Thu Dec 14 18:47:16 1989
Luid: root  Euid: root  Ruid: root  Egid: root  Rgid: root
Event type: Access denial
System call: Creat
Object:  /a
Result: Failed-EACCES (Access denial)
Security policy: discretionary
```

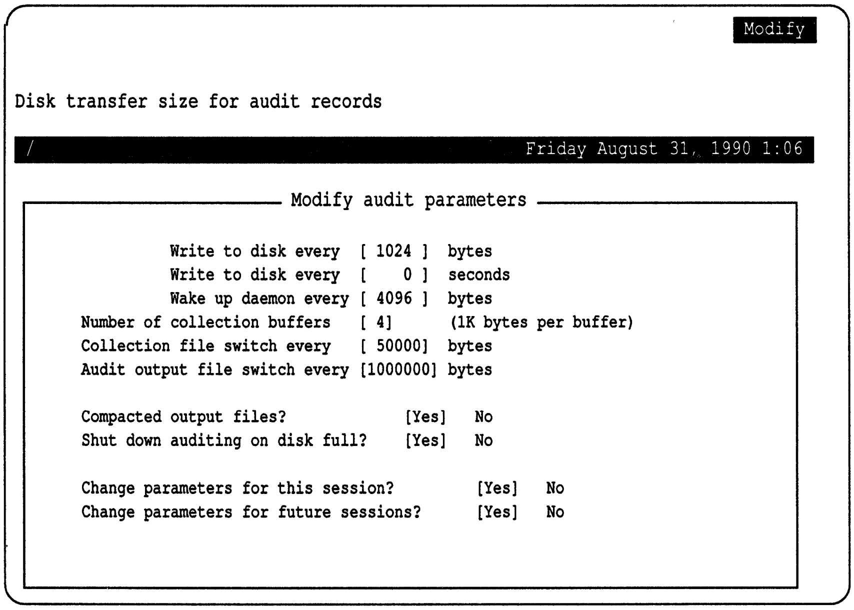
Note the (*INC*) next to the process ID. This indicates that the audit trail for this process is incomplete. It means that auditing was started *after* this user logged in, therefore there is no record of the LUID being set, and the reduction program doesn't know what it is. The reduction program assigns a value of zero (*root*) to any unknown LUIDs. The audit session must include the login for the user being examined, or the audit subsystem does not have a record of the user ID. The only way to ensure this is to start auditing before users are allowed to log in. You should avoid starting an audit session while the system is already active.

Adjusting audit performance parameters

You can alter some audit parameters to tailor auditing to the needs of a system. To examine the current audit parameter settings, make the following **sysadmsh** selection:

System ➪ Audit ➪ Collection ➪ Parameters ➪ Modify

A form similar to the following is displayed:



```
                                                                    Modify

Disk transfer size for audit records

 /                                                Friday August 31, 1990 1:06

                              Modify audit parameters

               Write to disk every  [ 1024 ]  bytes
               Write to disk every  [    0 ]  seconds
               Wake up daemon every [ 4096 ]  bytes
      Number of collection buffers  [ 4]      (1K bytes per buffer)
      Collection file switch every  [ 50000]  bytes
      Audit output file switch every [1000000] bytes

      Compacted output files?          [Yes]   No
      Shut down auditing on disk full? [Yes]   No

      Change parameters for this session?        [Yes]   No
      Change parameters for future sessions?     [Yes]   No
```

Some of these parameters relate to the earlier discussion on performance and reliability tradeoffs. This should become more apparent now. The parameters are as follows:

Write to disk every [ ] bytes
Write to disk every [ ] seconds

> These two parameters control the frequency with which audit data is written synchronously to the audit collection file from the internal audit buffers. Flushing can be controlled either by the amount of data that accumulates before writing or after a specific time interval. The latter is valuable when small amounts of data are generated and the frequency of the record generation is spread out over time. You can specify both byte count and time-lapse flushing. The time interval is always specified in seconds.

Performance may be adversely affected through a poor choice of either value. Writing too frequently slows the system with excessive I/O traffic. On the other hand, when these values are too large, the potential for data loss increases if the system crashes. A good rule of thumb is to flush each time a single internal buffer fills. Thus, setting the flush-byte count to 1024 (the size of an internal buffer) is usually sufficient.

Wake up daemon every [ ] bytes

This parameter controls the audit daemon. This daemon continually reads the audit device and retrieves records written to the collection files. These records are then compacted and written to compaction files that can later be reduced. To maximize the effectiveness of the compaction algorithm, the daemon needs to read blocks of data between 4 and 5 Kbytes. This requires special handling by the subsystem as a typical process read returns when any data is available rather than waiting for a specified amount of data to accumulate. For maximum effectiveness, this parameter should be left at the default value of 4 Kbytes. Values greater than 4 Kbytes will not yield significant improvement.

Number of collection buffers

This lets you specify the number of collection buffers for the subsystem to use. It uses these internal collection buffers to gather audit data for writing to the collection file. Multiple buffers are used to increase the efficiency of the system because all processes essentially share the buffer space attempting to write records. By providing multiple buffers, processes can deposit records and continue execution without blocking even if an I/O is occurring on previous buffers. A minimum of two buffers is required. Most systems cannot effectively use more than 4-6 buffers to avoid performance problems. There is no deterministic way to calculate the optimum number of buffers. Generally, base this value on the expected process load of the system.

Collection file switch every [ ] bytes
Audit output file switch every [ ] bytes

These two parameters let you specify the maximum size that collection and compaction files may grow before a new file is created. Choosing a small value for either parameter results in excessive file switches. Because compaction files are permanent, this can also lead to a proliferation of small files on the system. Choosing values that are too large creates a situation where audit collection files use large amounts of disk space even though they are partially read by the audit daemon and could otherwise be discarded.

The size of audit compaction files can be controlled because these files remain on the system until reduced and removed. It is desirable that these files be of reasonable size to work with, including being able to save and restore them easily. The default value for the collection files is 50 Kbytes, and the compaction files are 1 Mbyte. Make sure that the maximum size chosen for the compaction files does not exceed the **ulimit** established for the system, which determines the maximum size of a user file.

Compacted output files

This option is provided should non-compacted audit files be desired. There is no compelling reason why this option should be exercised because compaction does not require large amounts of additional processing time and the resultant disk savings are typically greater than 60 percent. The compaction algorithm is contained in the audit daemon user process, not performed in the kernel portion of the subsystem.

Enable audit on system startup

This option starts auditing automatically each time the system is rebooted. This field is only displayed with the View option; it is set according to whether auditing was enabled or disabled. If auditing was disabled, then auditing is disabled at startup.

Shut down auditing on disk full

This option allows the system to shut down automatically if the system runs out of disk space, thus avoiding data corruption.

Change parameters for this session
Change parameters for future session

The last two options on the screen let you dynamically alter the current session and/or make the changes a permanent part of the audit parameter file for future sessions.

File and directory management

This section explains how to choose the location of audit record files, back up and remove old records, and maintain the disk space used by the audit subsystem.

Audit data is divided into sessions, with a new session started each time auditing is stopped and started once again. When examining or manipulating audit data, it is done by session number.

The following session file functions are available:

List	list audit session files on the system
Backup	back up an audit file session to backup media
Delete	remove an audit session file
Restore	restore an audit file session from backup media

An audit session consists of a session log file and a group of compaction files generated between an enable and disable of the audit subsystem. Each collection file and compaction file created during a session is uniquely numbered with the session in which it was created. When sessions are completed, only the log file and the compaction files remain. The file maintenance functions examine which sessions are still on the system and let you remove sessions no longer wanted.

The audit directory maintenance functions are as follows:

List	display audit directory list
Create	create new audit directory
Delete	delete existing audit directory
Add	add entry to audit directory list
Remove	remove entry from audit directory list

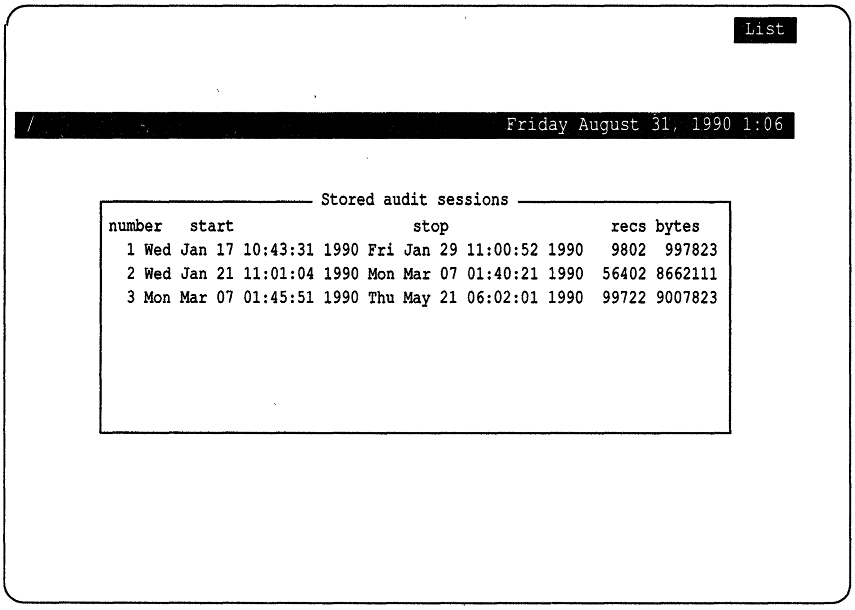
The last two functions manipulate the list of directory entries used by the audit system rather than the directories themselves.

Listing audit sessions

To list the audit sessions that are currently on the system, make the following **sysadmsh** selection:

System ➪ Audit ➪ Files ➪ List

A screen similar to the following is displayed:



```
/                                            Friday August 31, 1990 1:06

                  Stored audit sessions
number   start                     stop                       recs  bytes
  1 Wed Jan 17 10:43:31 1990 Fri Jan 29 11:00:52 1990   9802  997823
  2 Wed Jan 21 11:01:04 1990 Mon Mar 07 01:40:21 1990  56402 8662111
  3 Mon Mar 07 01:45:51 1990 Thu May 21 06:02:01 1990  99722 9007823
```

The number in the first column is the session number. This list is what is displayed when you ask for a list of session numbers from other options of the audit menus.

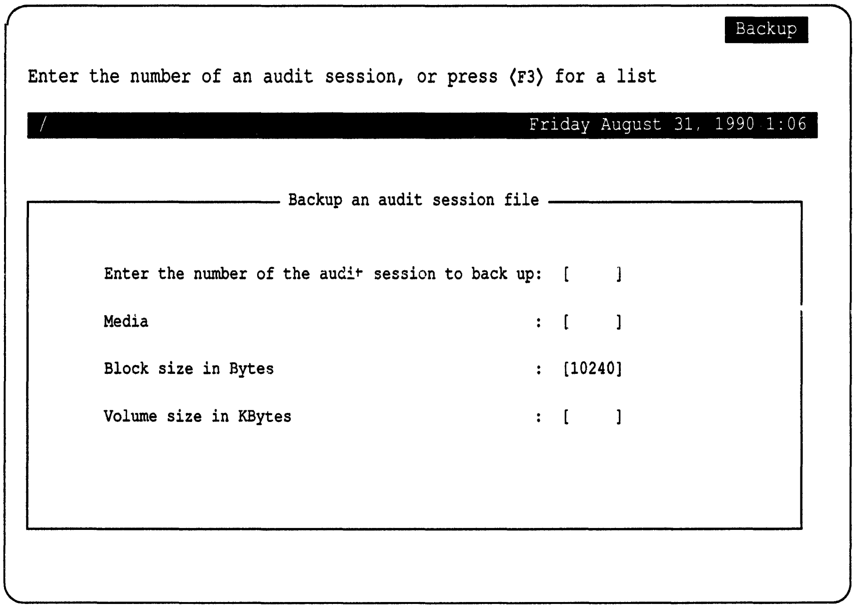
NOTE The session display is not a listing of session data files. It is the total number of records and bytes of that session; the actual data is stored in a series of collection files.

Backing up audit files

To back up audit files, make the following **sysadmsh** selection:

System ➪ Audit ➪ Files ➪ Backup

A screen similar to the following is displayed:



```
                                                                  Backup

Enter the number of an audit session, or press ⟨F3⟩ for a list

 /                                                  Friday August 31, 1990 1:06


  ─────────────────────── Backup an audit session file ───────────────────────


     Enter the number of the audit session to back up:  [     ]

     Media                                           :  [     ]

     Block size in Bytes                             :  [10240]

     Volume size in KBytes                           :  [     ]
```

Because audit sessions require a large amount of disk space, it is often necessary to archive audit data and either reduce it later or retain it for some period of time in case it is needed to analyze problems that are not immediately detected. The backup and restore interface provides this capability. The Backup option requires a session number as input. This can be obtained by pressing ⟨F3⟩ for a list of sessions. After selecting Backup, you must select an output device for the backup. This can be any removable media available on the system.

WARNING Auditing consumes a great deal of disk space. Depending on how many users on your system and how many events are audited, it may be necessary to back up and remove session files on a weekly basis. If you have scheduled backups, it is probably not necessary to use the audit backup selection. Again, it is important to remove the files to free disk space after they are backed up.

Restoring audit files

Similarly, sessions that were backed up onto removable media using the interface program can be reloaded using the Restore option. To do so, insert the media containing the saved session files into the restore device, and specify the device name.

To back up audit files, make the following **sysadmsh** selection:

System ➪ Audit ➪ Files ➪ Restore

You are asked to enter the device to restore from; press ⟨F3⟩ for a list.

Removing audit files

To remove audit files after backing them up, make the following **sysadmsh** selection:

System ➪ Audit ➪ Files ➪ Delete

The Delete selection is provided for the removal of audit sessions. Sessions can be archived to backup media and removed to make room on the filesystem for more audit files. Sessions are removed using the session number, which can be obtained by pressing ⟨F3⟩. The session number is then presented to the Delete option to delete all of the files associated with that session.

Monitoring disk space consumption

You can monitor the disk space consumed by the audit records using the **sysadmsh** selection:

System ➪ Audit ➪ Collection ➪ Statistics

The total count of audit data written is the number of bytes currently stored on the system for the current session.

Maintaining collection directories

Both collection files (generated by the subsystem) and compaction files (generated by the audit daemon) are written to directories you specify. An audit session may contain files written to many different directories. At the conclusion of a session, only the compaction files remain, because the collection files are removed by the subsystem as they are read by the audit daemon. You do not need to keep track of the directories into which files are written because a session log file maintains this information.

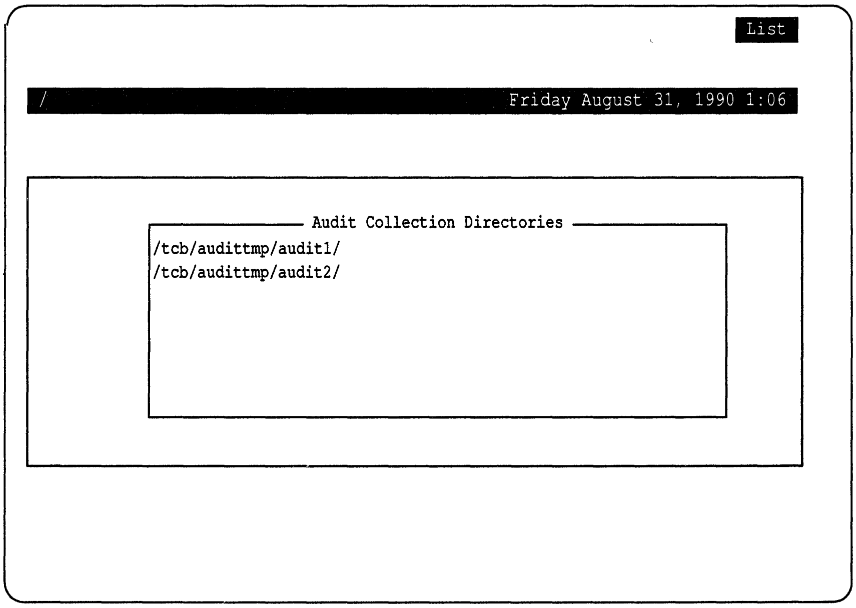
You can improve the system's performance by placing the audit directories on a filesystem that resides on a different physical device from the rest of the filesystems. This reduces contention for disk resources. Also, auditing requires large amounts of space, even with compaction. The subsystem warns you when disk space is low, and it eventually disables auditing if the free space of a filesystem is too low. For this reason, multiple directories are supported by the subsystem and the daemon. If an error occurs in writing to a directory or if space is exhausted, the subsystem and the daemon attempt to use alternate directories to continue.

Listing collection directories

To list the current audit collection directories, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Collection ⇨ Directories ⇨ List

A screen similar to the following is displayed:



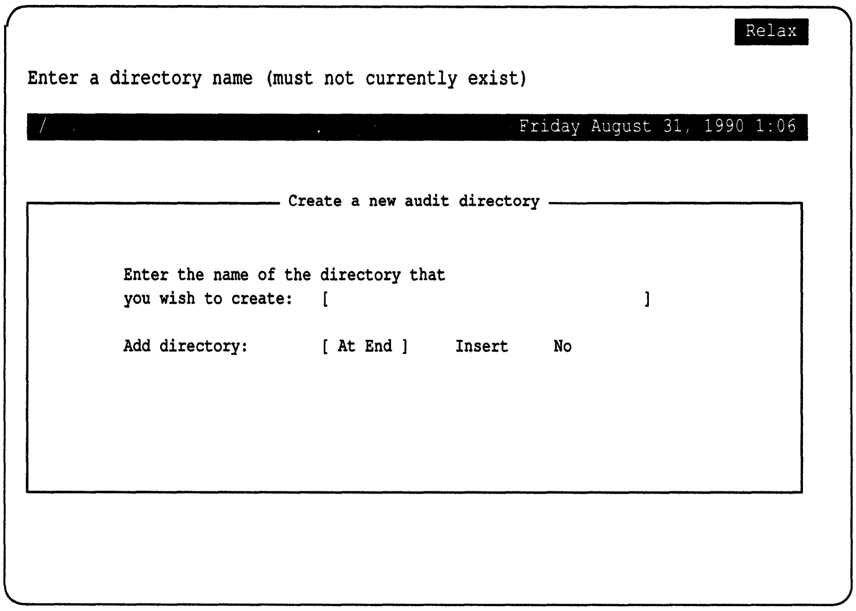
```
                                                          List

 /                                     Friday August 31, 1990 1:06


              ─────────── Audit Collection Directories ───────────
              /tcb/audittmp/audit1/
              /tcb/audittmp/audit2/
```

Creating a collection directory

To create a new collection directory (and optionally add it to the list used by the audit subsystem) make the following **sysadmsh** selection:

System ➪ Audit ➪ Collection ➪ Directories ➪ Create

A screen similar to the following is displayed:



```
                                                          Relax

Enter a directory name (must not currently exist)

/                                          Friday August 31, 1990 1:06

---------------------- Create a new audit directory ----------------------

         Enter the name of the directory that
         you wish to create:   [                              ]

         Add directory:        [ At End ]     Insert     No
```

You must supply the full pathname of the directory. The Add directory options are as follows:

At End	adds new directory at end of existing list
Insert	inserts new directory before an existing one
No	does not add this directory to the collection directories list

Enter each filename as an absolute pathname. There is no artificial limit on the number of directories you may specify. If no directories are specified, the subsystem and the daemon create all files in the root filesystem using the reserved audit subsystem directory */tcb/audittmp*, the default configuration file setup. Directories are used sequentially as they are filled with data; this is why it is necessary to specify the position. When session files are backed-up and removed from the audit directories, the system places new audit data in the first directory.

Deleting a collection directory

To delete an existing collection directory, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Collection ⇨ Directories ⇨ Delete

You are asked to select the directory to be deleted. Press ⟨F3⟩ for a list.

Adding a collection directory entry

You can also add an existing directory to the list used by the audit subsystem. Directories are used in the order listed. Thus, a new entry can either be inserted into the list or placed at the end. To add an entry to the collection directory list, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Collection ⇨ Directories ⇨ Add

You are asked to select the directory entry to be added to the list and specify the placement; select At End or Insert.

Removing a collection directory entry

To remove an entry from the collection directory list, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Collection ⇨ Directories ⇨ Remove

You are asked to select the entry to be removed. Press ⟨F3⟩ for a list.

Report generation

The reduction function uses a file called a *report template* to perform post-selection of audit records. This file is built by the audit administrator interface program based on your input. You can build and save multiple files, each with a different set of selection criteria. Reduction may then be run several times on the same session data with a different report template each time. Thus, you can build and save report templates used frequently in data reduction. When the actual data reduction is needed, you can use the files already built.

The following Report options are available:

List	lists all report templates available
View	views the parameters stored in a report template
Create	creates a new report template
Modify	modifies an existing report template
Delete	deletes an existing report template
Generate	makes a reduction run, specifying audit session and report template

As discussed previously, audit collection criteria represents the first level of audit selection. After the data is gathered, it can be further processed, or reduced, to generate a useful collection of data about a specific aspect of system operation. The data reduction menus let you select to reduce and determine what records are desired. The Generate option supports a wide range of post-selection criteria that helps you target specific events, users, or objects. This requires the session number and the report template, which may be any of the report templates built using the report template create or update options.

The options for List, View, Create, Modify, and Delete are used for report template maintenance.

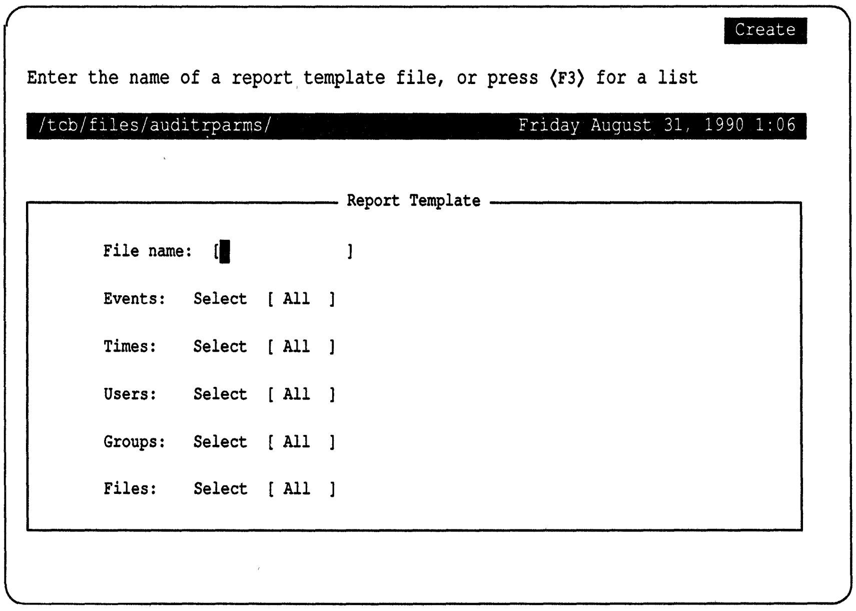
Creating or modifying a report template

To create or modify a report template, make one of the following **sysadmsh** selections:

System ⇨ Audit ⇨ Report ⇨ Create
System ⇨ Audit ⇨ Report ⇨ Modify

The screens for the Create and Modify options are identical. Modify allows you to take an existing report template and make any modifications desired.

A screen similar to the following is displayed:



```
                                                                  Create

Enter the name of a report template file, or press ⟨F3⟩ for a list

 /tcb/files/auditrparms/                          Friday August 31, 1990 1:06


 ---------------------------- Report Template ----------------------------

       File name:  [▌          ]

       Events:     Select  [ All ]

       Times:      Select  [ All ]

       Users:      Select  [ All ]

       Groups:     Select  [ All ]

       Files:      Select  [ All ]
```

Enter the name of the template file to be created or modified. Use ⟨F3⟩ to select from a list of templates; several are distributed with the system.

The selections are as follows:

Events	The audit events can be selected or all collected. Events not selected cause those records to be discarded from the output.
Times	The start and stop times for collection. If a security-related event was suspected between certain times of the day, you could use this feature to select those records that were generated during that time period. This could serve to concentrate the analysis on those records that are likely to reveal what has happened.
Users/Groups	Both users and groups of users can be singled out for audit. If a certain user account was the target of a penetration, you could select only those records that were generated from user or group IDs that matched that user. This permits the record search to be concentrated on suspected accounts.
Files	Files (object names) can also be used to select audit records from the output. For records that contain multiple object names, if a specified name matches *any* object in the record, the record is selected. The object names must be specified as absolute pathnames because all object names are resolved from relative to absolute names by the reduction program.

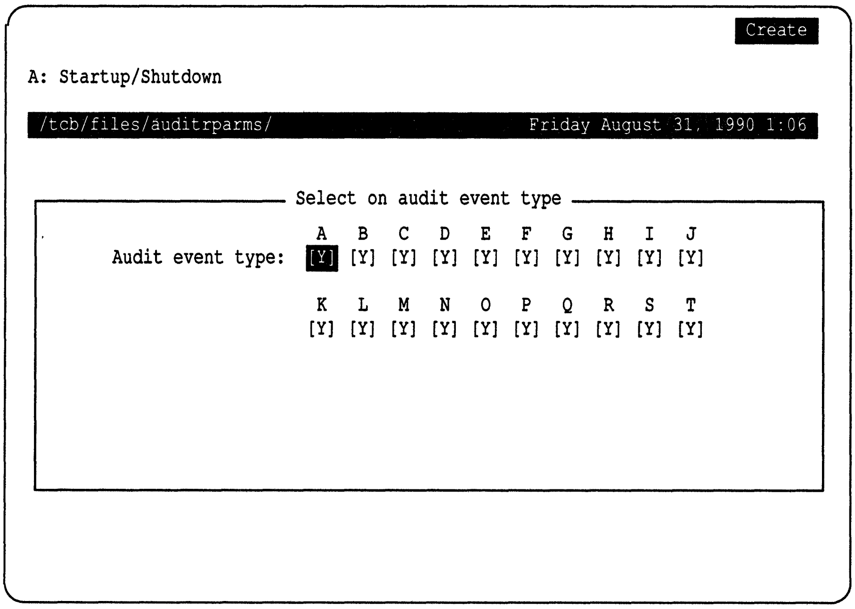
If All is chosen for a category, all events, times, and so forth are selected. If you choose Select, a window or form is displayed so that the desired criteria can be selected individually.

Any combination of the above criteria can be used. For instance, time interval, user ID, and object name can be combined for a single session. If a record is within the specified time interval that was generated by a selected user, and has one of the selected objects in the record, then it is selected for output.

There is a precedence for record selection that governs the combination of the selection criteria. If the audit event type is not specified, the record is not selected, regardless of other criteria. Likewise, if time stamp selection is enabled and the record does not meet the criteria, the record is not selected. If the record passes the selection criteria for event type and time, then the record is selected if it has a user ID (login, effective, or real), group ID (effective or real), or an object in the record that is specified in the report template. If no users, groups, and objects are specified, only event type and time selection is performed.

Events

The selection for Events is highlighted first. If you choose Select and press ⟨Return⟩, the following window opens to allow the events to be selected:

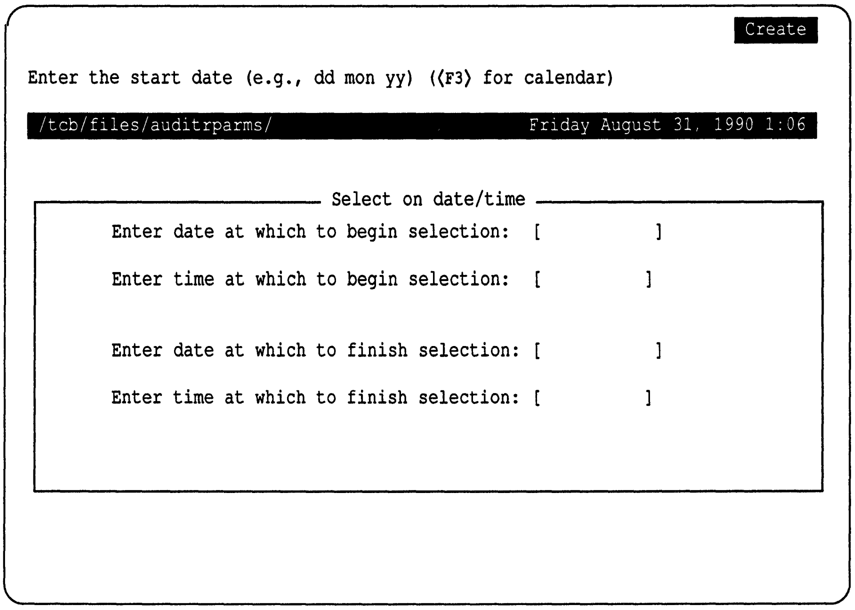


```
A: Startup/Shutdown

/tcb/files/auditrparms/                        Friday August 31, 1990 1:06

                   Select on audit event type
                                A   B   C   D   E   F   G   H   I   J
          Audit event type:    [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y]

                                K   L   M   N   O   P   Q   R   S   T
                               [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y] [Y]
```

Depending on the template you selected, the events will have a "Y" or "N" in brackets. To toggle an event from yes to no, use the ⟨Space⟩. When you are satisfied, press ⟨Return⟩ to save your changes.

Times

The Times selection is highlighted. Use the arrow key to highlight All, or Select and press ⟨Return⟩. If you choose Select, the following window opens to allow start and stop times to be selected:

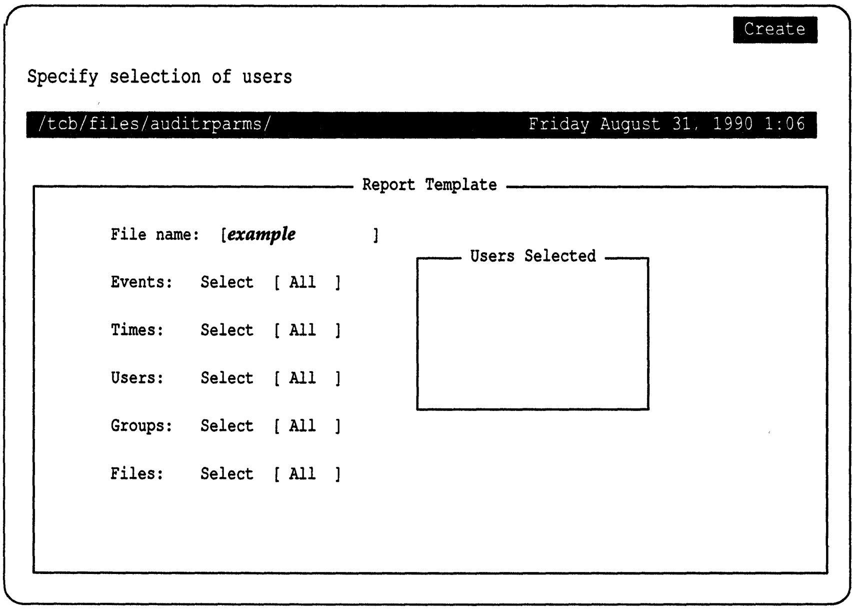


```
                                                          Create

Enter the start date (e.g., dd mon yy) (⟨F3⟩ for calendar)

 /tcb/files/auditrparms/                      Friday August 31, 1990 1:06

 ------------------------ Select on date/time -------------------------
|      Enter date at which to begin selection:  [          ]            |
|                                                                       |
|      Enter time at which to begin selection:  [        ]              |
|                                                                       |
|                                                                       |
|      Enter date at which to finish selection: [          ]            |
|                                                                       |
|      Enter time at which to finish selection: [        ]              |
|                                                                       |
 -----------------------------------------------------------------------
```

You are asked in turn to enter the date and time for beginning and ending selection. You can press ⟨F3⟩ to get a calendar.

Users

The Users selection is highlighted. Use the arrow key to highlight All, or Select and press ⟨Return⟩. If you choose Select, the following window opens to allow users to be selected:

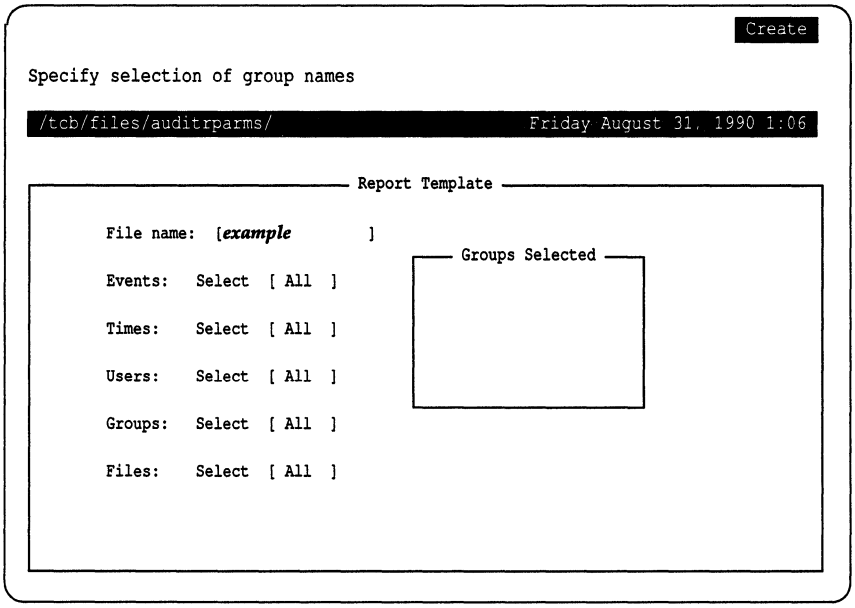


```
                                                              Create

Specify selection of users

 /tcb/files/auditrparms/                     Friday August 31, 1990 1:06

 ------------------------ Report Template -----------------------------

        File name:  [example        ]
                                          ---- Users Selected ----
        Events:   Select  [ All  ]

        Times:    Select  [ All  ]

        Users:    Select  [ All  ]

        Groups:   Select  [ All  ]

        Files:    Select  [ All  ]
```

You can modify the list of users and/or add new names to the list in the window. Press ⟨Return⟩ when the list is complete.

Groups

The selection for Groups is highlighted. Use the arrow key to highlight either All, or Select and press ⟨Return⟩. If you choose Select, the following window opens to allow groups to be selected:

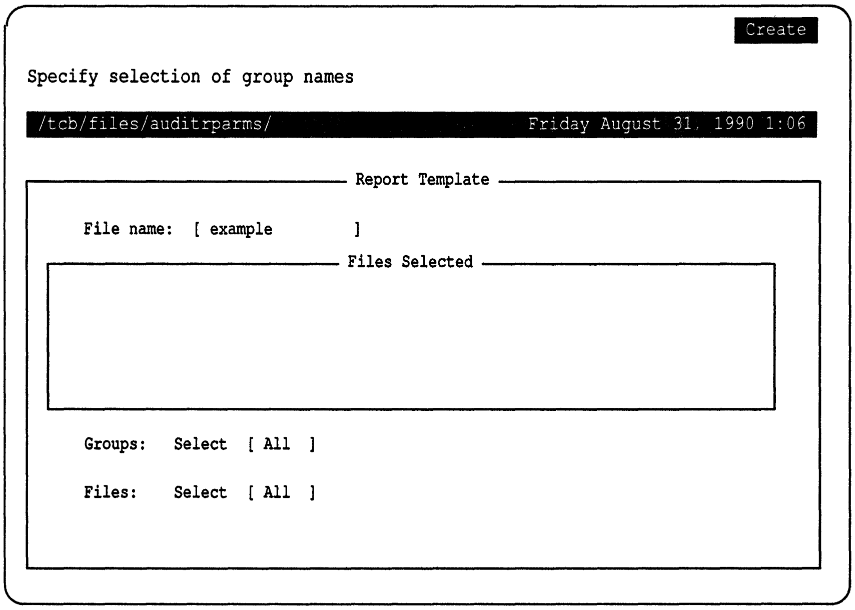


```
                                                              Create

Specify selection of group names

/tcb/files/auditrparms/                          Friday August 31, 1990 1:06

 ---------------------------- Report Template ------------------------------

      File name:  [example      ]
                                          ---- Groups Selected ----
      Events:   Select  [ All  ]

      Times:    Select  [ All  ]

      Users:    Select  [ All  ]

      Groups:   Select  [ All  ]

      Files:    Select  [ All  ]
```

You can modify the list of groups and/or add new names to the list in the window. Press ⟨Return⟩ when the list is complete.

Files

The final category is Files. If you specify the files (objects) here, it restricts the auditing of object deletion, modification, and so on to the files selected here. Use the arrow key to highlight either All, or Select, and press ⟨Return⟩. If you choose Select, the following window opens to allow files to be selected:



```
Specify selection of group names

/tcb/files/auditrparms/                          Friday August 31, 1990 1:06

                         Report Template

    File name:  [ example        ]
                          Files Selected



    Groups:    Select  [ All  ]

    Files:     Select  [ All  ]
```

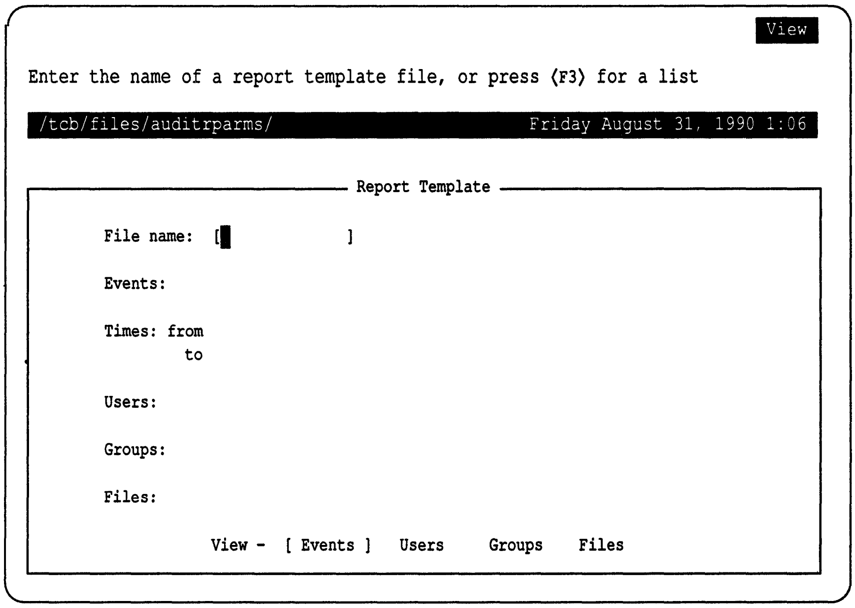
You can modify the list of files and/or add new names to the list in the window. Press ⟨Return⟩ when the list is complete.

Viewing a report template

To view a report template, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Report ⇨ View

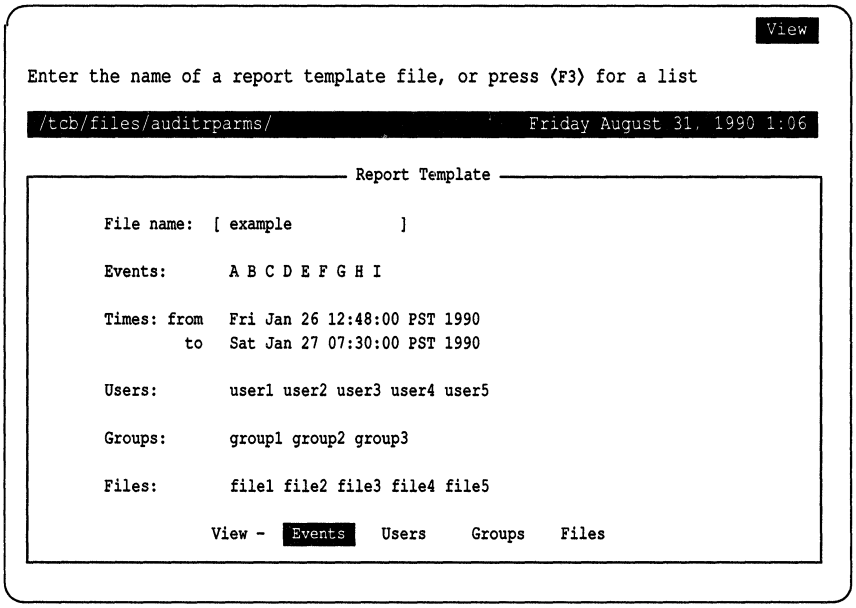
A screen similar to the following is displayed:



```
                                                                    View

Enter the name of a report template file, or press (F3) for a list

 /tcb/files/auditrparms/                        Friday August 31, 1990 1:06

                            Report Template

      File name:  [          ]

      Events:

      Times: from
               to

      Users:

      Groups:

      Files:

             View -  [ Events ]  Users    Groups    Files
```

Use the ⟨F3⟩ key to select a template to view. You cannot make changes through this selection; if you wish to customize a template, you should use the Modify selection instead.

Once a template is selected, the fields are filled in as in the following example:



```
                                                                      View

Enter the name of a report template file, or press <F3> for a list

/tcb/files/auditrparms/                           Friday August 31, 1990 1:06

─────────────────────────── Report Template ───────────────────────────

      File name:  [ example          ]

      Events:      A B C D E F G H I

      Times: from  Fri Jan 26 12:48:00 PST 1990
               to  Sat Jan 27 07:30:00 PST 1990

      Users:       user1 user2 user3 user4 user5

      Groups:      group1 group2 group3

      Files:       file1 file2 file3 file4 file5

                View -  Events   Users    Groups   Files
```

The selections at the bottom of the screen are used to open windows to display data for each category that does not fit on the screen.

Listing report templates

To list the available report templates, make the following **sysadmsh** selection:

System ⇨ Audit ⇨ Report ⇨ List

A list of the report templates available is displayed. The following are shipped with the system:

admin.actions	These are administrator actions; this is event type L.
all.objects	These are actions relating to the creation, modification, or removal of objects; these are event types D-I.
authorization	This is use of authorization; these are event types M and T.
dac.events	These are DAC (Discretionary Access Control) changes or denials; these are event types J and K.
denials	These are DAC denials; these are event types A-K, and N.
login.action	This is a record of logins (successful and unsuccessful) and logouts; this is event type B.

Removing report templates

To remove a report template, make the following **sysadmsh** selection:

System ➪ Audit ➪ Report ➪ Delete

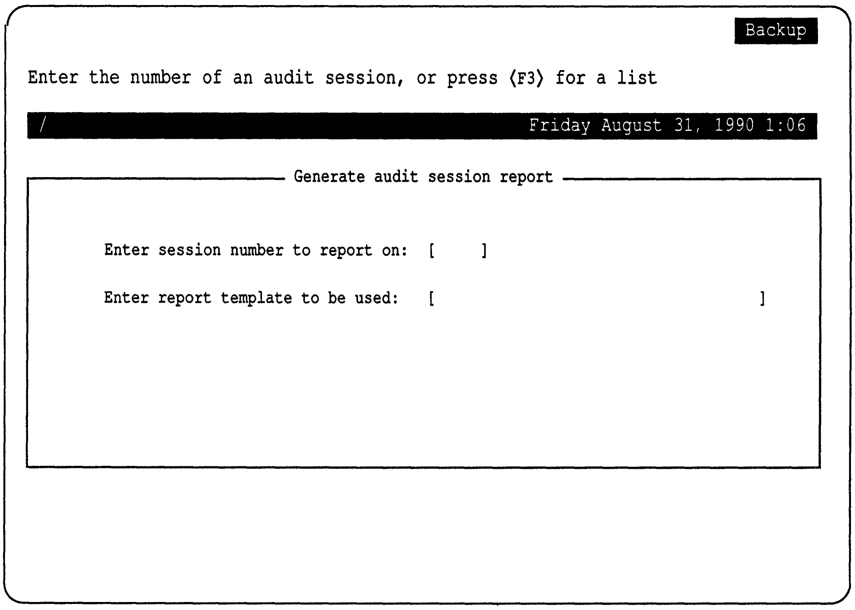
You are asked to provide the name of the report template to remove. Press ⟨F3⟩ for a list.

Generating an audit report

To generate an audit report, make the following **sysadmsh** selection:

System ➪ Audit ➪ Report ➪ Generate

A screen similar to the following is displayed:



```
Enter the number of an audit session, or press ⟨F3⟩ for a list

 /                                             Friday August 31, 1990 1:06

                  Generate audit session report

       Enter session number to report on:  [     ]

       Enter report template to be used:   [                              ]
```

Enter the session number or press ⟨F3⟩ for a list; do the same for the report template. You are then asked where to send the output, to the terminal, a file, or the printer. It is best to direct the output to a file. When the report generation begins, note that it may take some time if the volume of data is high. For example, if your report template does not specify dates and times for beginning and ending selection, the entire audit session is reduced, which could consist of tens of megabytes of data.

Example report and template

Example 12-3 is an example of an audit report based on a template with the following characteristics:

```
Events: B K M T
Times:  Start: Fri Feb 2 19:00  Stop: Fri Feb 2 21:00
Users:  johnp
Groups:None
Files:  All
```

The report template concentrates on undesirable activities, such as attempts to access restricted system files, running restricted administrative programs, and so forth. In this simplified example, user *johnp* logged on and attempted to remove (unlink) */etc/passwd*. In a real scenario, there would be more records to examine. This example serves to demonstrate the power of audit data. The next section "Understanding audit reports" is a detailed study of how audit information is interpreted.

Example 12-3 Audit report output

```
        ***** Audit Data Reduction Program *****

Audit session number:   2
Collection system name: unix
Collection file count:  15
Compaction file count:  1
Total audit records:    11034
Total uncompacted size: 696050
Total compacted size:   243262
Data compression rate:  65.05
Collection start time:  Fri Feb  2 19:00:15 1990
Collection end time:    Fri Feb  2 21:00:00 1990

        ***** Selection Criteria *****

Time Interval Selection:
        Start: Fri Feb  2 19:00:00 1990
        Stop:  Fri Feb  2 21:00:00 1990
Event Type Selection:
        Event type: Login/Logoff activity
        Event type: Access denial
        Event type: Insufficient privilege

UID selection in effect.

        johnp

        ***** Audit Records *****

Process ID: 235        Date/Time: Fri Feb  2 19:55:42 1990
Event type: Login/Logoff activity
Action: Successful login
Username: johnp
Login terminal:  /dev/tty01

Process ID: 267        Date/Time: Fri Feb  2 19:56:11 1990
Luid: johnp  Euid: johnp  Ruid: johnp  Egid: group  Rgid: group
Event type: Access denial
System call: Unlink
Object:  /etc/passwd
Result: Failed-EACCES (Access denial)
Security policy: discretionary

Process ID: 280        Date/Time: Fri Feb  2 19:58:14 1990
Event type: Login/Logoff activity
Action: Logoff
Username: johnp
Terminal:  /dev/tty01
```

Understanding audit reports

To interpret the audit trail, you need to understand the records produced by the program and what they mean. Remember that audit records come from three sources: system calls, trusted applications, and protected subsystems. Record formats differ greatly among these three sources. Further, system calls differ greatly from one another in content because of the specific function being performed. For instance, a process creation, **fork**(S), need only indicate the process ID of the newly created process and the ID of its spawning process (parent). However, for an **open**(S) system call, an object is being acted upon and the name of that object must be recorded. For system calls like **mount**(S) and **link**(S), still more information must be recorded; each requires that two object names be recorded. The reduction facility sorts records presented to it and outputs the information in an organized manner.

Output records can be classified into two types: system call records produced by the kernel audit mechanism and application audit records. Some items are considered common to all output records. For instance, the date and time of the record and the process ID associated with the record are printed for each type. Beyond this, the content of a record depends on what was audited.

System call record formats

System call records account for the majority of the records in the audit trail. The operating system contains over 60 system calls. Not all of these system calls are audited as only some of these are deemed to be security-related. Slightly over half of the system calls have the potential to create an audit record. Some system calls support multiple functions (such as **fcntl**(S), **msgsys**(S), **shmsys**(S), and **semsys**(S)) that may only generate audit records for certain functions. For instance, the **fcntl**(S) system call allows files to be opened by duplicating open file descriptors and also permits file-specific flags maintained by the kernel to be retrieved. The first case constitutes an auditable event, making an object available to a subject, while the second has no real security relevance. Furthermore, system calls may perform functions that are considered auditable events but are not enabled by the system event mask at the time.

For the most part, the output of system call records is the same for all calls. Variations exist because some system calls operate on objects (such as **open**(S)) and the object name is contained in the record. Each contains at least the time, date, process ID, system call name, event type, login user ID, real user and group IDs, effective user and group IDs, and an indicator of success or failure for the call.

Each output record contains these basic information fields and others depending on the system call. The basic record is shown in Example 12-4. This illustrates the common header along with the system call and result fields.

Example 12-4 Common output record header.

```
Process ID: 68        Date/Time: Sat Mar  5 13:25:09 1988
Luid: root  Euid: root  Ruid: root  Egid: root  Rgid: root
Event type:
System call:
Result:
```

Each system call is classified into a system event type based on the actions that are performed. This describes the event type of the system call. The actual system call name is given. In most cases this uniquely identifies the action. Unfortunately, some UNIX system calls are overloaded, causing a system call entry point to be used to accomplish multiple actions. For example, **msgsys**(S) is the system call entry for message queue IPC operations. This single entry point calls **msgget**(S), **msgop**(S), and **msgctl**(S) to perform certain IPC functions.

System calls like this are not self-explanatory. The audit subsystem is aware of these overloaded calls and provides additional information to identify the particular function. For system calls that succeed, the result is specified as successful. For each that returns an error, the error provides additional record classification. For instance, an **open**(S) that fails from lack of permission is classified as an access denial. An unsuccessful system call that generates an audit record indicates the error in the result field.

The system call output records can be divided into two groups. The first group contains records that do not require pathnames in the audit record. For instance, the **fork**(S) system call is audited to track new processes as they are spawned into the system, but the audit record does not require a pathname. On the other hand, **open**(S) returns a file descriptor for the specified pathname. Subsequent operations, like **close**(S), use the file descriptor. To provide meaningful audit records, this second type of record must contain the pathname. Using the reduction function, this pathname is associated with all further actions on that file, even though the action may have been performed with a file descriptor.

Figure 12-1 lists audited system calls that do not contain pathname information.

pipe	**fork**	**kill**	**close**
setuid	**setgid**	**exit**	**security**
read	**setpgrp**	**msg**	**dup**
sem	**shm**	**write**	**fcntl**

Figure 12-1 System calls without pathnames.

An output record from one of the above system calls uses the generic record mask described in Example 12-4. The following example illustrates the output record from a successful **setuid**(S) system call.

Example 12-5 setuid(S) system call record.

```
Process ID: 6381        Date/Time: Tue Mar 15 11:25:19 1988
Luid: blf  Euid: blf  Ruid: root  Egid: root  Rgid: root
Event type: Modify process
System call: Setuid
Result: Successful
```

Similarly, Example 12-6 shows the output record from a **setuid**(S) system call that failed due to a lack of permission on the file. Notice that the event type classification is different and that the error is reflected in the result field.

Example 12-6 Access denial output record.

```
Process ID: 6381        Date/Time: Tue Mar 15 11:25:19 1988
Luid: blf  Euid: blf  Ruid: blf  Egid: guru  Rgid: guru
Event type: Modify process
System call: Setuid
Result: Failed (EPERM)-Not owner
```

Many system calls in this group generate additional information in the output record to help clarify the audit trail. The semaphore, shared memory, message queue and **security**(S) system calls are overloaded. They map to multiple functions. These audit records identify the specific function being performed and also the affected object (for example, shared memory). **close**(S), **dup**(S), and **fcntl**(S) operate on file descriptors that were mapped from pathnames. An output record indicating a **dup**(S) of a file descriptor would not be very useful because it does not uniquely identify the file. Thus, the file descriptor correlates to a pathname and prints the pathname in the record.

Even though the **read**(S) and **write**(S) system calls are listed in Figure 12-1, they are audited only in certain circumstances and neither has a dedicated output record. Both system calls are audited only for the first occurrence for a file. Subsequent reads and writes do not need to be audited as they provide no additional information. The audit records are used to track the state of the file. When the file is closed due to **exec**(S), **exece**(S), **close**(S), or **exit**(S), the name of the object and an indicator of whether the file was read or written is included in the system call record for the action that caused the file to be closed. This is illustrated in Example 12-7.

Example 12-7 close(S) system call record.

```
Process ID: 421       Date/Time: Sat Mar  5 17:15:09 1988
Luid: blf  Euid: blf  Ruid: blf  Egid: guru  Rgid: guru
Event type: Make object unavailable
System call: Close
File Access-Read: Yes  Written: No
Object: /tmp/datafile
Result: Successful
```

The second group of system calls, shown in Figure 12-2, contains pathnames as part of the output record. The pathname represents the target of the system call. Two of the system call records actually contain two pathnames: **link**(S) and **mount**(S).

open	**unlink**	**creat**
exec	**chdir**	**mknod**
chown	**chmod**	**stat**
umount	**exece**	**chroot**
link	**mount**	

Figure 12-2 System calls with pathnames.

Each of the system calls in Figure 12-2 takes one or more pathnames as arguments to the call. The pathnames are audited and become an important part of the reduction process. Output records for these calls indicate the object name acted upon. This name is also retained by the reduction facility and, where applicable, is associated with the file descriptor returned by the system call. This provides a mapping for other system calls like **dup**(S) that operate on the file but do not contain the pathname. Example 12-8 shows an output record generated from a **creat**(S) system call. The record format is the basic format augmented by the pathname.

Example 12-8 Output record with pathname.

```
Process ID: 64       Date/Time: Sat Mar  5  23:25:09 1988
Luid: root  Euid: root  Ruid: root  Egid: root  Rgid: root
Event type: Object creation
System call: Creat
Object:  /tmp/daemon.out
Result: Successful
```

All of the calls in this group use the same format for pathnames. Two calls, **link**(S) and **mount**(S), operate on two pathnames: a source and a target. Both names are audited and reflected in the output record by the reduction facility. A typical record produced by a **link**(S) system call is shown in Example 12-9.

Example 12-9 Output record with two pathnames.

```
Process ID: 14231     Date/Time: Thu Mar 16  03:25:39 1988
Luid: lp  Euid: lp  Ruid: lp  Egid: lp  Rgid: lp
Event type: Object creation
System call: Link
Source:  /tmp/printfile
Target:  /usr/spool/lp/lp3014
Result: Successful
```

Two other records in this group generate special output records. These are **chown**(S) and **chmod**(S), which are used to alter discretionary access permissions and file ownership for objects. Due to the security-relevant nature of their actions, the previous and new values of the object owner, group, and mode are output in the record. Example 12-10 illustrates the output record from a **chmod**(S) system call.

Example 12-10 chmod(S) system call record.

```
Process ID: 6841      Date/Time: Sat Mar  5 13:25:09 1988
Luid: blf  Euid: blf  Ruid: blf  Egid: guru  Rgid: guru
Event type: Discretionary Access Change
System call: Chmod
Object:  /tmp/demo/newfile
Old values: Owner-blf  Group-guru  Mode-100600
New values: Owner-blf  Group-guru  Mode-100666
Result: Successful
```

Application audit records

There are six different types of audit records generated by application programs. The formats for these are similar. Unlike system calls, any record produced in one of the six categories is always formatted identically, although the information varies. The categories are:

- login and logoff events
- user password events
- protected database events
- audit subsystem events
- authorized subsystem events
- terminal and user account lock events

Each record contains some information common to all audit output records. This includes the process ID, the time and date, and the audit event type. The remainder of the output record depends on the record type. The record-specific fields are described in the following sections.

Login/Logoff record

All attempts to log into the system are audited by the **login** program. This is true of successful as well as unsuccessful attempts. This creates an important trail of user accesses to the system and also a trail of attempted accesses. You can use the audit records for login or logoff to determine who actually used the system. It is also valuable in determining if repeated penetration attempts are being made. The operating system supports the option of locking terminals after a certain number of attempts and this event can also be audited. Thus, you have all tools necessary to monitor (and prevent) access to the system.

Each login record contains an indicator of the specific action that was audited. The three possibilities are: successful login, unsuccessful login, or logoff. All successful logins and logoffs result in an audit output record that indicates the user account and terminal of the login session. For unsuccessful attempts, the user name is meaningless, because the attempt failed. In this case, only the terminal on which the attempt occurred is output along with the basic record fields. Example 12-11 illustrates the output from a successful login.

Example 12-11 Successful login audit output record.

```
Process ID: 2812      Date/Time: Fri Mar  4 10:31:14 1988
Event type: Login/Logoff Activity
Action: Successful login
Username: blf
Terminal: /dev/tty2
```

User password record

All attempts, successful or not, to modify a user account password are carefully audited by the authorization subsystem. To avoid revealing user passwords, audit records for these events contain no password text, but only indicate the account and action that was audited. The actions are classified into successful password change, unsuccessful change, and lack of permission to change the password. Example 12-12 shows an audit record for an unsuccessful password change.

Example 12-12 Unsuccessful password change audit record.

```
Process ID: 7314      Date/Time: Tue Mar  1 18:30:44 1988
Event type: Authentication database activity
Action: Unsuccessful password change
Username: blf
```

Protected database record

Programs that maintain and modify the system's protected databases audit all access attempts and unusual circumstances associated with the databases. This may range from integrity problems to security-related failures. In addition to the record header and the specific audit action, the output includes the name of the program detecting the problem, the object affected by the problem, expected and actual values, and the action and result of the event. See Example 12-13.

Example 12-13 Protected database output record.

```
Process ID: 7314      Date/Time: Tue Mar  1 18:30:44 1988
Event type: Authentication database activity
Command: authck
Object: Protected password database
Value: Expected-0 Actual-0
Security action: /tcb/files/auth/code
Result: extraneous file in protected password hierarchy
```

Audit subsystem record

Events that affect the operation of the audit subsystem itself are audited very carefully. The **sysadmsh** audit selections and the audit daemon, **auditd,** both generate audit records for functions they support. Additionally, the audit device driver also writes audit records for certain function requests. The functions audited include the following:

- subsystem initialization
- subsystem termination
- subsystem parameter modification
- audit daemon enabled
- audit daemon disabled
- subsystem shutdown
- subsystem error

Each output record includes the common header information along with an indicator of the function audited. This provides an accurate accounting of all attempts to affect the operation of the audit subsystem. Example 12-14 shows an actual audit record written to indicate the startup and initialization of the subsystem.

Example 12-14 Audit subsystem output record

```
Process ID: 517      Date/Time: Wed Mar  2  8:30:04 1988
Event type: Audit subsystem activity
Action: Audit enabled
```

Protected subsystem record

Each protected subsystem can generate audit records through the audit daemon. These records indicate unusual conditions detected by the subsystem. For instance, if a subsystem encounters permission problems with a file or is denied service due to lack of memory or some other resource, the subsystem generates an error message to that effect. You can use these records to help maintain the security and availability of the system.

Aside from the normal record header output, the subsystem records contain a subsystem name, an action, and a result. The subsystem name is the subsystem that detected the inconsistency and wrote the audit record. The action and result describe the action taken by the subsystem and the problem detected. Example 12-15 shows a subsystem-generated audit record.

Example 12-15 Authorized subsystem audit output record.

```
Process ID: 2812      Date/Time: Fri Mar  4 10:31:14 1988
Event type: Authorized subsystem activity
Subsystem: System Administrator Subsystem
Security action: Update /etc/rc
Result: Cannot open for update
```

Terminal and user account record

User accounts or terminals may become locked if the number of unsuccessful login attempts, as stored in the Authorization database, is exceeded. For instance, if a terminal is used to enter the system and the result is a series of unsuccessful logins, the **login** program may lock the terminal after a specified number of tries. Similarly, if a user attempts to log in to an account and fails repeatedly, that user account may be locked. Locking accounts and terminals prevents further access until the system administrator clears the lock. A terminal or user account lock may signal an attempted penetration of the system. These audit records contain the usual header information along with an identifier of the user account or terminal.

Example 12-16 User account lock output record.

```
Process ID: 517      Date/Time: Wed Mar  2  8:30:04 1988
Event type: System administrator activity
Action: User account locked by system administrator
Username: root
```

Auditing capabilities extended to users

It is possible to extend some auditing functions to users. You can allow users to generate audit reports of their own activities. The **audittrail** secondary subsystem authorization permits access to a subset of audit functions under System ➪ Audit ➪ Report. Report output is limited to records matching the user's LUID. Users can use all report selections, including the creation of report templates, which are stored along with the system templates.

Assignment of the **audittrail** authorization is discussed in "Changing user authorizations" in the "Administering user accounts" chapter of this guide.

Glossary of audit terms

An *audit collection file* is a file written by the audit subsystem device driver containing the raw audit data from all audit sources on the system, including system calls, trusted applications, and authorized subsystems.

An *audit compaction file* is a file written by the audit daemon containing buffers of data read from the audit device driver. The data can be in either a compacted or non-compacted format, depending upon options selected at the time the audit session was started.

The *audit daemon* is a daemon process started when the system makes the transition to multiuser state. It reads the audit subsystem device to retrieve audit records, compacts these records, and writes them to a permanent compaction file for later reduction. The daemon also acts as an interface program that permits non-protected subsystems to write audit records to the audit device.

An *audit session* is the period of time from audit enable until audit disable. During this time, the audit data is stored in compaction files written by the daemon. Each session is uniquely numbered and each file that is part of the session contains this unique ID in the filename. A master file is used for each session to collect session information and session file names for later reduction.

The *audit subsystem* consists of the components that provide the trusted audit services. This includes the audit device driver, the kernel audit mechanism, the audit daemon, the audit administrator interface, and the audit reduction facility.

An *audit trail* is the collection of audit data records from an audit session that can be reduced into a report of system activity.

audit reduction is the transformation of raw audit trail data into output records containing dates, user IDs, filenames, and event types. The output record describes the audited event in a readable text form.

configaudit is the kernel authorization that allows the audit parameters to be set for all users of the system.

The *event control mask* is the user-specific mask maintained in the Protected Password database on a per-user basis. This mask controls whether the user event mask prevails over the system default event mask when auditing is enabled. Each bit set in the control mask causes the event disposition mask to take precedence.

The *event disposition mask* is the user-specific mask used in conjunction with the event control mask for user audit event control. If the user event control mask has a bit set on, the corresponding bit entry in the event disposition mask determines whether the event is always audited or never audited. This holds true regardless of the system default event mask value.

An *event type* is a classification for each audit record. Security-related events on the system are classified into certain types that can be used to control audit generation or reduction. Every system action, regardless of success or failure, can be classified into an event type. This event type then determines the disposition of the record.

An *object* is an entity acted upon by a subject (such as files, shared memory segments, semaphores, pipes, or message queues).

post-selection is the selective use of collected audit data. Post-selection involves collecting audit data for all events and users so the audit trail is as complete as possible. Any security-related event is in the audit trail compaction files at the end of a session.

preselection selectively controls audit record generation. This allows certain users and events to generate audit records while others are discarded. The result is a more compact audit trail with less detail than if full auditing was used.

report templates are generated through the administrative interface to control the selective reduction of audit sessions. Selective criteria control the user, object, and event selection for output records.

A *subject* is an active entity that performs actions on objects, such as a process on the system that accesses files.

suspendaudit is a kernel authorization that suspends auditing.

The *system audit mask* is the default system event mask used to determine what events are audited when a user process mask does not take precedence.

The *user audit mask* collectively refers to the event control and event disposition masks that, together with the system default mask, control the generation of audit records on a per-process basis.

writeaudit is a kernel authorization that allows specific information to be recorded by the audit trail.

Chapter 13
Using MS-DOS and other DOS operating systems

Many users received MS-DOS, or other closely compatible DOS operating systems with their computer. This chapter explains how you can still use DOS utilities, files, and applications after you install the UNIX system. You can even access DOS files and directories on your UNIX system, or mount DOS filesystems and access the files directly. The UNIX system provides this facility so that you do not need to throw away your investment in DOS software, or buy another computer just to run a UNIX system.

Several programs make this coexistence possible. The **dos**(C) utilities allow access to DOS files on diskettes or on the DOS partition on the hard disk (provided the partition is unmounted). These utilities are discussed later in this chapter. The utility that partitions the disk is called **fdisk**(ADM) and is available in DOS and UNIX system versions. The next section explains how to use **fdisk** to create a DOS partition and a UNIX system partition on the same hard disk. Another section discusses installing a UNIX system partition on the hard disk along with DOS. There is also a section explaining various booting configurations, for users who mostly use the UNIX system and for users who mostly use DOS.

OS/2 coexistence

Although it may install successfully, OS/2 may not be bootable on your machine, regardless of whether a UNIX system partition is present or not; we cannot guarantee that OS/2 will work with your UNIX system. Refer to your computer's hardware documentation to determine if your machine is supposed to run OS/2. If you wish to use OS/2 and/or DOS on the same disk with your UNIX system, you must install them on the disk in the following order:

1. DOS Primary Partition
2. DOS Extended Partition
3. UNIX system
4. OS/2

NOTE If a second disk is installed you get a primary partition on the second disk and optionally an extended partition. You cannot boot DOS from the second extended partition.

There are no OS/2 tools available in UNIX System V (such as the DOS utilities described in this chapter). In addition, you must use **fdisk**(ADM) to switch to or from OS/2.

UNIX system **fdisk**(ADM) displays an OS/2 partition as DOS.

Partitioning the hard disk using fdisk

Each version of **fdisk** is documented in the respective operating system's manual. Unless otherwise noted, this chapter refers to the UNIX system version of **fdisk**(ADM).

fdisk is interactive, and uses a menu to display your options. Here is the main **fdisk** menu:

```
    1. Display Partition Table
    2. Use Entire Disk For UNIX
    3. Use Rest of Disk for UNIX
    4. Create UNIX Partition
    5. Activate Partition
    6. Delete Partition

Enter your choice or 'q' to quit:
```

The **fdisk** utility allows you to set up separate areas (partitions) on your hard disk for your operating system. The hard disk is divided into *tracks*. The number of tracks depends upon the size of the hard disk. A *partition* consists of a group of tracks. One hard disk may contain up to four partitions.

The **fdisk** command allows you to specify one disk partition as "active". This means that when you turn on (boot) your computer, the operating system installed in the active partition will start running. The UNIX system partition must be active when you intend to use your UNIX system.

You can also specify the number of tracks assigned to each partition. The number of available tracks will vary according to the size of your hard disk. Consult your Installation Guide for the recommended UNIX system partition size. The size of the UNIX system partition also depends on the number of software packages you want to install. You can install the UNIX system in this space, and have the rest of the space for user files and other software packages. Refer to the **custom**(ADM) manual page for information on how to install and remove software.

fdisk allows you to specify where the partition begins and ends. **fdisk** will not allow you to construct overlapping partitions. You do not need to install your UNIX system in the first partition.

You should always start your DOS partition at the beginning of the disk, starting at cylinder 0 or cylinder 1.

If you install a UNIX system partition on the same disk after the DOS, or extended DOS partition, start the UNIX system partition at the beginning of the next cylinder on the disk. To find the beginning of the next cylinder, note the ending track number of your DOS partition and start the UNIX system partition on the next track number that is a multiple of the number of heads on your hard disk. For example, if you have five heads on your hard disk and your DOS partition ends at track 103, start your UNIX system partition at track 105.

When you are running your UNIX system, the device name of the UNIX system partition is */dev/hd0a*. For more information about hard disk device names, see the **hd**(HW) manual page.

One option of **fdisk** tabulates the current state of the partitions (the Display Partition Table option). This option lists, for each partition, whether the partition is active, the first track, the last track, the number of tracks used, and the associated operating system. If you enter the Display Partition Table option and press ⟨Return⟩ to see the partition table, the result will be similar to Example 13-1.

Example 13-1 Sample fdisk table

```
Current Hard Disk Drive:  /dev/rhd00
```

Partition	Status	Type	Start	End	Size
1	Inactive	DOS	005	398	393
2	Inactive	DOS (ext)	400	1219	819
3	Active	UNIX	1220	2220	1000

```
Total disk size:  2229 tracks (9 tracks reserved for
masterboot and diagnostics).
```

Switching operating systems

There are three ways to switch to the primary DOS partition once you have set up separate DOS and UNIX system partitions:

- Enter **dos** at the boot prompt,
- Use a floppy diskette that contains the files necessary to boot the DOS operating system, or
- Use **fdisk** to change the current active partition.

When you use the boot prompt or a floppy to boot DOS, the UNIX system partition remains active even though you have switched operating systems. When you use **fdisk**, the UNIX system partition is inactive until you switch back to it.

To use the boot prompt method, enter:

dos

at the boot prompt:

```
SCO System V/386

Boot
:
```

NOTE The system boots from the first DOS partition found.

To use a floppy diskette to boot DOS, follow this procedure:

1. Make sure all users are logged off the system.
2. Run **shutdown**(ADM) to shut down the UNIX system. This command makes sure all users know the system is being shut down, terminates all processes, then halts the system.
3. Once the UNIX system has shut down, insert the bootable DOS diskette into the primary (boot) drive.
4. Boot DOS.
5. To get back to the UNIX system partition, remove any disks from the floppy drive(s) and press ⟨Ctrl⟩⟨Alt⟩-⟨Del⟩, or the reset key, or turn the computer off, then on. Since the UNIX partition is still active, your UNIX system boots.

Remember that if you have an active UNIX system partition and boot DOS from a floppy you can transfer to C: to work with the DOS files.

The other way to change operating systems is to run **fdisk** and change the active partition from the UNIX system partition to DOS. Then, after you shut down the system (see the previous steps) DOS boots from the hard disk. From here you can switch operating systems from the DOS partitions. You do not need a bootable DOS floppy disk as long as DOS is loaded on the DOS partition of the hard disk.

To switch back to the UNIX system partition, run **fdisk** under DOS and make the UNIX partition active. To reboot the UNIX partition, press ⟨Ctrl⟩⟨Alt⟩-⟨Del⟩, or the reset key, or turn the computer off, then on.

Because the UNIX system partition must be active for it to operate, you cannot use a bootable floppy to boot the operating system. This second method is appropriate for an occasional change of the active operating system.

Table 13-1 DOS hard disk devices

XENIX device convention	UNIX device convention
/dev/hd0d	/dev/dsk/0sd (linked with 0sC)
/dev/rhd0d	/dev/rdsk/0sd (linked with 0sC)
/dev/hd1d	/dev/dsk/1sd
/dev/rhd1d	/dev/rdsk/1sd

The hard disk device names in Table 13.1 are similar to */dev/hd0a* (the active disk partition) in that the disk driver determines which partition is the DOS partition and uses that as *hd0d* and *hd1d*. (You can use the XENIX or UNIX system device name conventions; they are equivalent.) This means that software that is running from the UNIX system partition and using the DOS partition does not need to know which partition is DOS (the disk driver determines that).

Installing a UNIX system partition on a DOS system

If you wish to set up your UNIX system on a hard disk which previously contained only DOS, follow these steps:

1. Copy (back up) all the DOS files and directories on the hard disk onto floppies, or whatever backup media you wish to use.
2. Run **fdisk**, under DOS. If there is enough free space for a UNIX system partition on your hard disk (check your *Installation Guide*), skip to Step 4. Otherwise, delete the DOS partition, then recreate it, leaving enough room on the disk for your UNIX system distribution and any other software that you intend to install.
3. Return the DOS files from the backup media to the newly created DOS partition on the hard disk. Keep the backups in case there is an error of some kind, so you will not lose any data.
4. Turn off your computer.
5. Follow the installation procedure outlined in the *Installation Guide* to install your UNIX system distribution.

   You will see a message warning that the contents of the hard disk will be destroyed. There is no cause for concern, because you have already backed up the DOS files and transferred them to the new DOS partition. The new partition being created will contain your UNIX system, and the installation process will only write information on the UNIX system partition.
6. During the installation procedure, **fdisk** is invoked to partition the hard disk. Use **fdisk** to assign a sufficiently large UNIX system partition.
7. Designate "UNIX" as the active operating system by choosing the "Activate Partition" option under **fdisk**.
8. Finish installing the UNIX system distribution.

NOTE UNIX **fdisk** displays DOS partitions as *DOS* while DOS **fdisk** displays UNIX system partitions as Other.

You can only create DOS partitions using DOS **fdisk**, and UNIX system partitions using UNIX system **fdisk**.

Be aware that DOS **fdisk** reports sizes in terms of cylinders, while UNIX **fdisk** reports sizes in terms of tracks. Check your hard disk manual for the number and size of cylinders on your hard disk.

Using a UNIX system and DOS with two hard disks

Your computer always boots the operating system in the active partition on the first hard disk. The UNIX system must boot from the first hard disk. There are several ways to configure your system if you have two hard disks and want to boot DOS. Two ways are discussed here.

One configuration consists of designating the entire first disk as a UNIX system partition. You then use a DOS boot floppy to start DOS and specify:

```
A> D:
```

to switch to the DOS area on the second hard disk, where **D:** is the designation for the second hard disk. This strategy works for some versions of DOS. Early versions recognize only the first hard disk on the system.

NOTE If you devote a hard disk for use with DOS, the disk must already be configured under DOS. See the "Adding hard disks and CD-ROM drives" chapter of this guide for details regarding hard disk configuration.

Another method is to maintain a small DOS partition on the first hard disk. The DOS partition is designated the active partition. In this configuration, the computer always boots DOS. This requires changing the active partition to boot the UNIX system from the hard disk.

If you use the entire second disk for DOS, you need only run **mkdev hd** to create device files for the second disk if you plan to use the UNIX system DOS utilities (**doscp, dosls, doscat,** and so on). If you do not wish to use those utilities to access DOS files on the second hard disk, there is no need to run **mkdev hd**.

NOTE Be sure to make a backup copy of your boot floppies if you use them to boot your secondary operating system.

Removing an operating system from the hard disk

You may find that you no longer need one of the operating systems installed on your hard disk. If you want to delete an operating system, use the appropriate version of **fdisk**. To delete a UNIX system partition, you must use the UNIX system version of **fdisk**. To delete a DOS partition, use **fdisk** under DOS. To delete an Extended DOS partition, you must delete all logical drives on that partition using **fdisk**. Deleting the partition removes the contents of that partition and leaves unallocated space.

You can then reallocate that space by either adding another UNIX system or DOS partition, or enlarging an existing partition. Enlarging a partition requires reinstalling the operating system and (for a UNIX system partition) remaking the filesystem on the partition using **divvy**(ADM).

DOS accessing utilities

The DOS accessing utilities are discussed in detail in "Using DOS accessing utilities" in the *User's Guide*. Note that you must have a bootable, although not active, DOS partition on the hard disk or a DOS floppy in order to use these UNIX system commands. For example, you can only transfer a file from a UNIX system partition on hard disk to a DOS floppy if either the DOS floppy is bootable or there is also a DOS partition on the hard disk.

You may also be able to use the UNIX system **dd**(C) and **diskcp**(C) commands to copy and compare DOS floppies. The UNIX system **dtype**(C) command tells you what type of floppies you have (various DOS and UNIX system types).

Also, the file */etc/default/msdos* describes which DOS filesystems (for example, A:, B:, C: . . .) correspond to which UNIX system devices.

NOTE You cannot execute (run) DOS programs or applications from your UNIX system.

The UNIX system does not record bad tracks in the DOS area of the hard disk. If a bad track develops in the DOS area, an operation such as **doscp** that attempts to access the affected area may fail. If such is the case, the message "Error on fixed disk" is displayed.

NOTE When trying to use the DOS utilities to access files on your DOS partition, you may see the error message "bad media byte". This message indicates that the DOS partition on the hard disk is not bootab!^ .'ou can make your DOS partition bootable by first backing up the files on the DOS partition, booting DOS from the floppy, and formatting the DOS partition using the command:

format c: /s

You should now reinstall your DOS files.

File and directory arguments

The file and directory arguments for DOS files take the form:

device:name

where ***device*** is a UNIX system pathname for the special device file containing the DOS diskette or DOS partition, and ***name*** is a pathname to a DOS file or directory. For example,

/dev/fd0:/john/memos

indicates that the file *memos* is in the directory */john,* and that both are in the device file */dev/fd0* (the UNIX system special device file for the primary floppy drive). Arguments without *device:* are assumed to be UNIX system files.

User configurable default file

For convenience, the user configurable default file */etc/default/msdos* can define DOS drive names that you can use in place of UNIX system special device file pathnames. For example, you can include the following entries in the above file:

```
A=/dev/fd096ds15
B=/dev/fd048ds9
C=/dev/dsk/0sC
D=/dev/dsk/0sD
```

Once you have defined the variables, you can use the drive letter **A:** in place of the special device file */dev/fd0* (96ds15 by default) when referencing DOS files or directories. For example:

/dev/fd0:/john/memos

can be replaced with:

A:/john/memos

The drive letter **B:** refers to a low density (48ds9) primary floppy drive. Drive letter **C:** refers to the primary DOS partition on the primary hard drive. **D:** refers to a logical drive in the extended DOS partition.

NOTE If you get the message "cannot open */dev/dsk/0sC*", or a similar message, check the user permissions on the special device file involved. As super user, change the permissions with the **chmod** command. For example:

```
chmod 666 /dev/dsk/0sC
```

gives full read and write permissions to all users for the special device file */dev/dsk/0sC*, which is the DOS partition on the primary hard disk.

Mounting DOS filesystems on a UNIX system

In addition to the DOS utilities provided with the Operating System to manipulate DOS files, (described in the *User's Guide*) it is also possible to mount a DOS filesystem and access its files freely while still operating from your UNIX system.

This means that DOS files can be edited or examined in place, without first copying them into the UNIX filesystem. The major restriction is that DOS files and applications cannot be executed under this arrangement; this requires use of VP/ix (if running under your UNIX system) or booting of the DOS partition. However, data files and text files can be examined, copied or edited.

NOTE On a mounted DOS filesystem you cannot use the DOS utilities. If used, they return an error message:

```
dosdir: FAT not recognised on /dev/dsk/0sC
```

You cannot create DOS filesystems using the **mkfs**(ADM) command. The DOS mounting feature is intended for existing DOS filesystems (as in floppy disks and an existing DOS partition).

Configuring support for mounted DOS filesystems

In order to mount DOS filesystems, the support for these features must be present in the kernel. If it is not, you must first add this to your kernel with the **mkdev**(ADM) command. Make certain you are logged in as *root* and enter the following command:

```
mkdev dos
```

Δ **sysadmsh** users select: System ➪ Configure ➪ Kernel ➪ DOS

This command adds the necessary functionality and prompts to relink the kernel. (If the link kit is not installed, you will be asked to install it.) After rebooting, you can mount DOS filesystems as described in the sections that follow.

How DOS filesystems are accessed

The operating system deals with DOS filesystems by superimposing certain qualities of UNIX system filesystems over the DOS filesystem without changing the actual files. UNIX system filesystems are highly structured and operate in a multiuser environment. Thus they include many distinctions that have no meaning under DOS, including:

- File ownership
- Access permissions
- Special files (pipes, device files, etc.)
- Links

NOTE Other applications/operating systems permit the mounting and access of DOS filesystems in this manner. However, most of them modify the DOS filesystem in some way to accomplish this. In the interests of portability, there are no proprietary modifications or extensions to the DOS filesystem. The ability to mount these filesystems is achieved purely through the facilities of the filesystem switch (FSS).

In order to make DOS files readily accessible, access permissions and file ownership are superimposed on the DOS filesystem when mounted.

Using the mount command

NOTE Make sure that you have run **mkdev dos** before attempting to use the **mount** command.

The form for a DOS filesystem mount command is:

mount -r -f DOS /dev/dsk/*xsy* /*mountpoint*

where:

x	is the hard disk number
y	is the drive letter (C:, D:, etc.)
mountpoint	is the name of the directory in the *root* filesystem where the DOS filesystem is to be mounted.

The **-r** flag mounts the filesystem read-only, an optional precaution that will prevent damage to the DOS filesystem, which is not as robust as a UNIX system filesystem.

NOTE DOS automatically calls the primary DOS drive, on the first disk, C:. If you have a primary DOS partition on the second disk this becomes D:, automatically, and logical drives on extended partitions are named in order, for example: disk0 Primary C: EXT E: F: G: H:, disk1 Primary D: EXT I: J:, etc. The naming convention in UNIX System V, for the above example, is as follows:

DOS	C:	D:	E:	F:	G:	H:	I:	J:
UNIX	0sC	1sC	0sD	0sE	0sF	0sG	1sD	1sE

When using **mount**, you must give the specific hard disk and partition numbers (as opposed to using wildcards).

Mounting a floppy disk

You can also mount DOS floppy disks, as in the following example using the 96-tpi floppy mounted on */mnt*:

```
mount -r -f DOS /dev/fd096 /mnt
```

Repairing and checking DOS filesystems

The operating system includes a DOS version of the **fsck**(ADM) utility that works on DOS filesystems. This utility reconciles the DOS FAT (File Allocation Table) to the files contained on the filesystem. When **fsck** is invoked, it automatically detects the DOS filesystem and invokes the proper binary.

Who can access the mounted DOS filesystem

Only *root* can mount a filesystem. Access by users is governed by the permissions and ownership that *root* places on the DOS filesystem. Because of the limitations discussed earlier, DOS does not recognize permissions or ownership. When mounted on a UNIX system, the DOS files behave as follows:

- The permissions and ownership of the filesystem are governed by the mountpoint. For example, if *root* creates a mountpoint */x* with permissions of 777, all users can read or write the contents of the filesystem. If the mountpoint is owned by *root*, all files within the DOS filesystem and any created by other users are all owned by *root*.
- The permissions for regular files will be either 0777 for readable/writable files or 0555 for read-only files. This preserves the consistency of the DOS filesystem. If a user can access the filesystem, the user will be limited by the permissions available under the DOS directory structure. This permission is read-only or read-write. When a file is created, the permissions are based on the **umask**(C) of the creator. For example, assume the user's **umask** is 022, which generates files with permissions of 777. Here are further examples.

Example 1: Creating a file. The permissions are based on the **umask** owner section. A umask of 022 will provide a file of 777 on the DOS partition. This is because the owner has not masked off the write bit for themselves.

Example 2: Examining a file already on the DOS partition. The permission you see is the logical AND of the UNIX system mountpoint permission and the DOS file permission. So, a UNIX system mountpoint of 750 and a DOS file permission of 555 will give you 550 for the permissions. This has nothing to do with the **umask**.

- There can only be one link for each file under the DOS filesystem. "`.`" and "`..`" are a special case under this arrangement and are not links as they are on a UNIX system.
- On UNIX systems, features such as locking govern how, under certain programs and applications, a file is accessed simultaneously by different users. These features operate identically on a mounted DOS filesystem. Two users can edit the same file and write to it as permitted by the locking mechanism used.

Appearance of DOS files

As no attempt is made to change the nature of DOS files, the carriage return character (^M) will be visible when editing a DOS file on a UNIX system. (UNIX systems use only a newline, while DOS uses a carriage return and a newline.) The **dtox**(C) and **xtod**(C) commands are the easiest way to switch the end-of-line format. **dtox** is used to change DOS format to UNIX system format, and **xtod** vice-versa. These tools are described in more detail in "Using DOS accessing utilities" in the *User's Guide.*

Restrictions

Additional logical restrictions that must be observed relating to filenames, modification times, and backup utilities are as follows.

File names

The rules for file names and their conversion follows the guidelines found in the **dos**(C) manual page. In addition, the standard DOS restrictions on illegal characters apply. However, wildcards can be used just as they can with a UNIX system.

Modification times

When accessed from the UNIX system partition, the creation, modification, and access times of DOS files are always identical and use GMT, or Greenwich Mean Time. (This is because UNIX System V uses GMT internally and converts it for the user.) This means that files created in the DOS filesystem while under a DOS or UNIX system will not have consistent times across the operating systems.

UNIX backup utilities

The **backup**(ADM) and **xbackup**(ADM) utilities cannot be used to make backups of a mounted DOS filesystem. DOS utilities and other copy programs like **tar**(C) will work as expected.

For more information, including more technical aspects of DOS usage, refer to **dos**(C).

UNIX systems and DOS on non-Standard disks

The UNIX system provides support for "non-standard" hard disks. The term "non-standard" refers to hard disks for which there are no correct disk parameter entries in your computer's ROM.

The correct parameters you specify for your non-standard disk(s) are stored in the masterboot block, which is the first sector of your boot hard disk drive. The hard disk characteristics are specified during UNIX system installation and these characteristics are then written out with the rest of the masterboot block. The special masterboot block that comes with your UNIX system distribution resets the disk parameters to the specified values no matter which operating system is "Active." This mechanism provides nonstandard disk support for both UNIX and DOS systems.

Although the special masterboot supports nonstandard disks under DOS, you cannot use your UNIX system to install DOS on your hard disk. If a nonstandard disk is being used, it is assumed that you already have some method to transfer your DOS files to the hard disk.

Unless you are changing the active partition, you should only use the UNIX system **fdisk** to manipulate your hard disk partition table. Using DOS **fdisk** or custom **fdisk** provided by hard disk manufacturers after the UNIX system has been installed may disable nonstandard disk characteristics, rendering your disk inaccessible.

Chapter 14
Administering serial terminals

The most important aspect of a multiuser system is the addition and maintenance of serial terminal devices. Adding terminals lets more users access the system and adds to overall system capabilities.

This chapter explains the following tasks:

- enabling serial terminal devices for operation, including serial multiscreens
- maintaining serial terminals
- configuring and using **mscreen**(M), a serial terminal equivalent of **multiscreen**(M)
- configuring and using scancode-compatible terminals

A prerequisite to adding terminals to your computer can include the addition of physical ports to your system. Consult the "Adding multiport cards, memory, and other bus cards" chapter of this guide to install the card, using the **mkdev serial** command to prepare the ports for use.

Adding a serial terminal

Before you add a serial terminal to your system, look in the hardware manual for your terminal for instructions on connecting the terminal to a serial line. Also, refer to the list of standard serial lines in the "UNIX directories and special device files" chapter of this guide to find the name of your serial line. (If you add a serial card, the possible names of the additional device files are listed in **serial**(HW) or in the documentation for cards that include driver software.)

Many types of terminals are supported. Look in the **terminals**(M) manual page for a comprehensive list of terminals supported. Support for terminals is provided through the **terminfo**(M) database, which contains the definitions and classifications of keystrokes and control sequences that vary from terminal to terminal. For a description of the *terminfo* database, see the **terminfo**(M) and **terminfo**(F) manual pages.

The following steps show how to install a terminal with the standard COM serial lines or with serial expansion cards:

1. If you are adding a terminal directly to a COM port, you need not run **mkdev serial**. Otherwise, you should consult "Adding multiport cards, memory, and other bus cards" to configure the ports for terminal connections.
2. Make sure you are logged in as *root* in multiuser mode.

   Plug in your terminal and turn it on. Set it for 9600 baud, 8 data bits, 1 stop bit, no parity, full duplex, and XON/XOFF handshaking. If your terminal does not work in this mode, look for advice on configuring your terminal in the section "Changing the gettydefs File" later in this chapter and in the **stty**(C) manual page.

   Some terminals connect with a straight cable directly to the computer. Other terminals connect to a modem. Terminals connected to a modem use a "null modem" or "modem connector," which is a cable with pins 2 and 3 crossed. Connect the terminal so that Transmit Data on the serial port is connected to Receive Data on the terminal, and Transmit Data on the terminal is connected to Receive Data on the serial port. Signal Ground should be connected to Signal Ground. Other pins probably do not need to be connected. The operating system requires only that pins 2, 3, and 7 are connected.

   Enable the terminal with the **enable**(C) command.

   For more information on your terminal, refer to your terminal manual or a reference on serial communication.

3. Check that the entry for this serial port in the */etc/inittab* file looks like the following (***ttyname*** is the name of the device file, for example */dev/tty1a*):

   ```
   t1a:2:respawn:/etc/getty tty1a m
   ```

   The */etc/inittab* entry should appear as above. If the entry does not look like this example, edit the file to correct it. Information on the format of the */etc/inittab* file can be found in the **inittab**(F) manual page. The last field in the */etc/inittab* entry is the line mode, a label which corresponds to an entry in the */etc/gettydefs* file. (**gettydefs**(F) specifies serial line characteristics, including baud rate.) In the example above, **m** corresponds to the **m** entry in */etc/gettydefs*. Although the UNIX system serial driver does not support baud rates exceeding 9600 baud, */etc/gettydefs* entries are provided for use with third-party serial drivers that do. These entries are labeled **n** and **o** and specify a baud rate of 19200 and 38400 baud, respectively.

   WARNING If you make any changes to */etc/inittab* by hand that you wish to be permanent, you must also make the same change to */etc/conf/init.d/sio*. This is because each time the kernel is relinked (as when a driver is added or a tunable parameter is changed), */etc/inittab* is reconstructed from the entries found in */etc/conf/init.d/sio*.

4. If the port is enabled, press the ⟨Return⟩ key a few times to see if a "login:" prompt appears. If so, you are ready to log in. If not, use the console or a working terminal to log in as the super user (*root*), and disable the port with this command:

   disable ***ttyname***

   where ***ttyname*** is the device special name of the port in question. Make sure you are using a non-modem control device, for example, */dev/tty1a*, not */dev/tty1A*.

5. From the console, as *root*, see if you can redirect output to the terminal by entering:

   date > /dev/*ttyname*

   If you do not see the date printed on the terminal and you are not sure of the correct ***ttyname***, try other ttynames on that serial port. If you still do not see the date printed on the terminal, then try the following:

   - Make certain that the terminal is plugged in.
   - Check that the cable is configured correctly. If the serial port you are using has a 25-pin connector (DB-25), read through step 2 in the preceding set of instructions. Are pins 2, 3, and 7 connected correctly? (Note that pins other than 2, 3, and 7 are probably not used.)

     If your system or expansion card has a 9-pin connector (DB-9), you must use a 9-pin to 25-pin connector. Look in your hardware manual for information on 9-pin to 25-pin connections.

- Check your terminal setup configuration. See step 2 in the preceding set of instructions. Try changing the baud rate.
- Check the switches on your serial port. If you are using a multiport card, try other lines on that card.
- Attach the terminal to a standard serial port (COM1 or COM2) to see if the terminal and cable are working correctly. If you are already using a COM port, try switching to another one.

  If you have successfully installed another terminal, switch hardware between the working and the nonworking terminal one piece at a time. This may help you isolate a hardware problem. Note that some faulty hardware may work under DOS but not on a UNIX system.

6. When the date prints on your terminal, enable the port with the following command:

   enable ***ttyname***

   The **enable** command starts a **getty** process that displays the following login prompt:

   ```
   login:
   ```

   If you do not see the "login:" prompt, enter the following command to verify that **getty** is running on the port and that the software is configured properly:

   ps -t ***ttyname***

   Your screen should display a message similar to the one in the following example, with either "login" or "getty" listed in the "COMMAND" column:

   ```
   PID     TTY     TIME    COMMAND
   2557    1a      0:06    getty
   ```

7. If you have typed the **enable** and **disable** commands many times, it is possible that a new **getty** cannot be spawned on that port. If so, shut the system down, reboot, log in as *root* in multiuser mode, and try again.

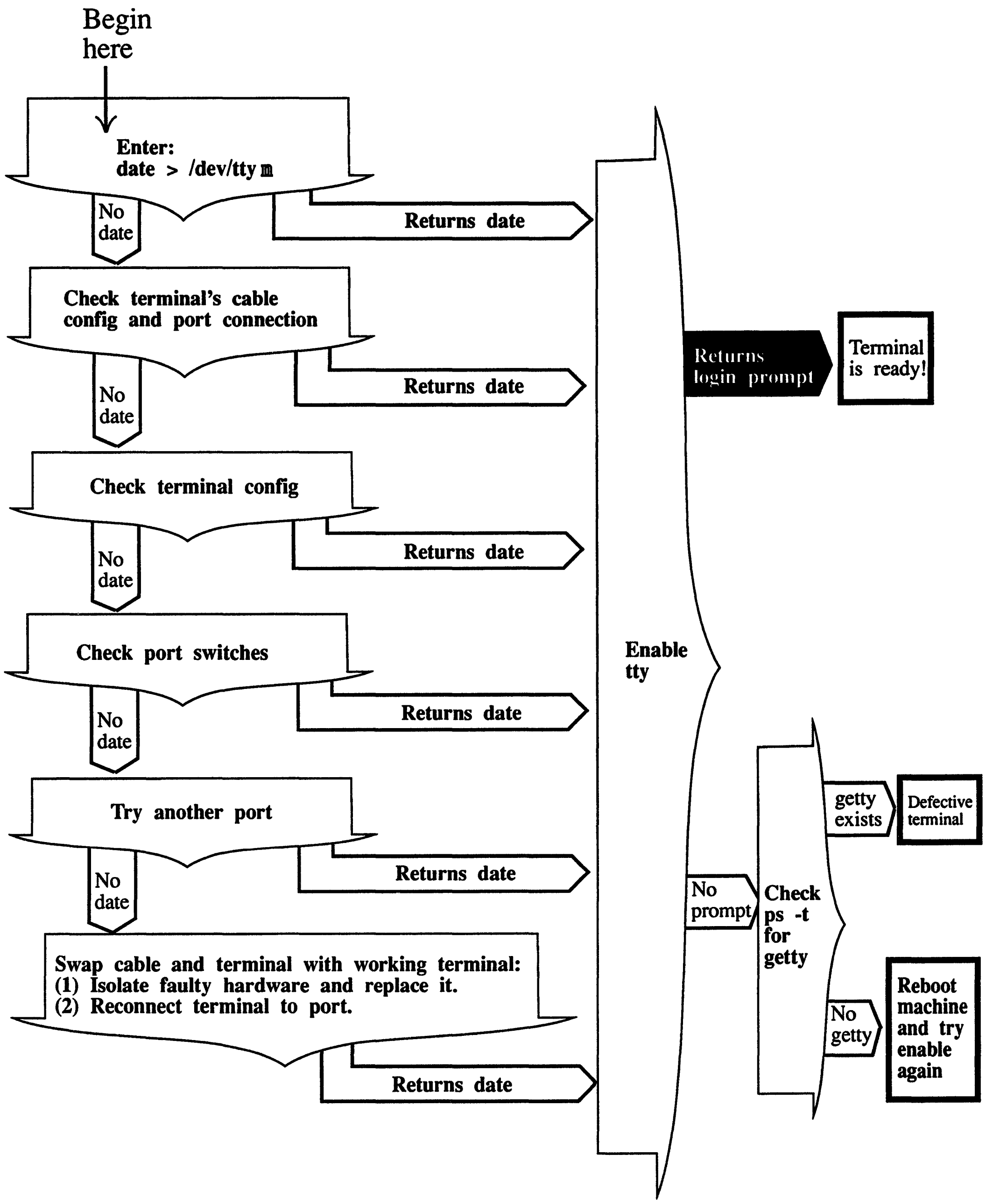


Figure 14-1 Checking the terminal connection

Changing default terminal line characteristics

Your system can automatically adapt to several different terminal baud rates and settings. The same program that displays the login message, **getty**(M) (for "get tty"), reads these terminal line values from a table, trying each setting until one is successful, and the user can log in to the system. This table provides several default settings for different kinds of terminal lines.

getty automatically executes as part of the login process. The table of terminal settings is found in a file called */etc/gettydefs*. You can edit *gettydefs* to add different sets of terminal characteristics or to change the existing ones.

The gettydefs file

The file */etc/gettydefs* contains the information that **getty** uses to set up terminal line characteristics such as baud rate. The file is in the form of a table. Each table entry is divided into five fields and one optional field. These fields include:

label # initial-flags # final-flags # login-prompt # next-label

The fields are:

label	identifies the *gettydefs* entry to **getty**. This could be a number or a letter. The label held corresponds to the line mode field in */etc/inittab*. **init** passes the line mode to **getty** as an argument.
initial-flags	sets terminal line characteristics when **getty** first establishes the connection. **getty** recognizes the flags listed in the **termio**(M) manual pages. Often the only flag in this field is the one setting the baud rate. For example, B300 would set the speed to 300.
final-flags	sets the terminal line characteristics just before **getty** executes **login**. These flags describe the operating characteristics for the line. The baud rate (B) is set again. Other common flags include **SANE** (a composite flag that sets a number of terminal characteristics to reasonable values), **TAB3** (expands tabs with spaces), **IXANY** (enables any character to restart output), and **HUPCL** (hangs up line on final close). Flags can be entered in any order.

login-prompt contains the login message that greets users. This field is printed exactly as it is entered, including spaces and tabs. An "@" symbol in the login-prompt field is expanded to the first line (or the second line if it exists) in the file */etc/systemid* (unless the "@" is preceded by a "\").

Several character sequences are recognized, including:

\n	Line feed	\t	Tab
\r	Carriage return	\f	Form feed
\v	Vertical tab	\b	Backspace
nnn	(3 octal digits) The specified ASCII characters		

next-label identifies the next label in *gettydefs* for **getty** to try if the current one is not successful. **getty** tries the next label if a user presses the ⟨BREAK⟩ key while attempting to log in to the system. Groups of entries, such as dial-up or TTY lines, should form a closed set so that **getty** cycles back to the original entry if none of the entries is successful.

Each field is separated by a number sign "#", and each entry in *gettydefs* is separated by a blank line.

Here is the default entry from */etc/gettydefs* for terminal lines:

```
m # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # m
```

Here is a description of each part of this line:

- The letter "m" identifies this entry to **getty**.
- The next field sets the baud rate to 9600.
- The third field indicates the baud rate (B9600), and several line characteristics including **SANE** (a composite flag for a number of characteristics), and **HUPCL** (hangs up line on final close).
- "\r\n@!login:" is expanded to display the system name in the login: prompt.
- The last field is the ***next-label***. In this example, the label "m" directs **getty** to repeat this entry if it is unsuccessful. In the case of dialup lines, these labels can be used to direct **getty** to cycle through a series of lines (for example, 300-1200-2400).

Changing the gettydefs file

The file */etc/gettydefs* has sets of entries for the dial-up lines and terminal lines. These different sets correspond to line-mode settings in */etc/inittab*. The **init** program passes the line mode as an argument to **getty**.

You can edit *gettydefs* to add new terminal settings or to change existing ones. For example, the settings for terminal lines might look like the following:

```
4 # B2400 HUPCL # B2400 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 5
5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 6
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 4
```

To change the sample *gettydefs* file so that the first baud rate **getty** attempts is 1200, do the following:

1. Enter a text editor to edit the first line of the file *gettydefs*.
2. Change the first and third fields from B2400 to B1200.
3. Save *gettydefs* and exit the editor.

The sample file should look like the next example:

```
4 # B1200 HUPCL # B1200 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@|login:# 5
5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@|login:# 6
6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login:# 4
```

You can also add additional terminal line settings to *gettydefs*. Flags and permissible values for terminal settings are listed in the manual page **stty**(C).

When you add a new entry, be sure that the groups of entries in *gettydefs* form a closed set, so the next-label field of the last entry directs **getty** back to the first entry in the group.

To add an entry for a baud rate of 300 to the preceding sample *gettydefs* file, follow these steps:

1. Enter a text editor to edit the file */etc/gettydefs*.
2. Locate the point where you want to insert the new settings for *gettydefs*. The order of the entries does not matter; **getty** only looks for the label. In this example, the new entry is the last entry in the file.
3. Insert a carriage return after the last line in the file and enter the following on a new line:

   ```
   7# B300 HUPCL # B300 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 4
   ```

4. To incorporate label 7 into the set of labels, change the next label field for entry 6 to 7:

   6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login:#7

   getty is now directed from label 6 to 7, and then back to 4. Make certain that a blank line separates each pair of entries.

5. Exit the text editor, saving the revised *gettydefs* file.

The new *gettydefs* looks like the following:

```
4 # B1200 HUPCL # B1200 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 5

5 # B4800 HUPCL # B4800 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 6

6 # B9600 HUPCL # B9600 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 7

7 # B300 HUPCL # B300 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 4
```

Checking the terminal settings

Each time you change the terminal line settings or add new entries to *gettydefs*, you should check to make sure that the new values make sense to **getty**. To do this, you use the command **getty** with the check option, **-c**, and the filename.

For example, to check *gettydefs*, enter:

```
getty -c /etc/gettydefs
```

The file is scanned and the results are displayed. If any of the values and settings in *gettydefs* are not permitted, **getty -c** reports them. For more information on **getty** and *gettydefs*, see the **getty**(M) and **gettydefs**(F) manual pages.

Changing serial line operation

Whenever you enable a terminal with the **enable** command, the system automatically sets the operating characteristics of the serial line to a set of default values. Sometimes these values do not match the values used by the terminal and, therefore, must be changed to allow communication between the system and the terminal. You can display the operating characteristics of a serial line with the **stty** (for "set tty") command. If you need to change the characteristics of a port that is enabled, you should use the entires in the *gettydefs* file rather than the **stty** commands given below.

NOTE Any settings on a port using the **stty** command only last as long as the port in question is still open. As an example, if you want to change the baud rate of tty2a, and tty2a is not enabled, the **stty** command first opens the port, then changes the port settings, and finally closes the port. When it closes the port for the last time, the settings revert to the original. In the **stty** commands later in this section, the use of the **while** loop is to avoid this behavior of **stty**. If you run **stty** redirecting input without **< /dev/*ttyname***, it works on your current serial line, which you have open. In this case, because the serial line stays open after the **stty** command, the settings also stay in place.

You can display the current operating characteristics of a serial line by entering this command at the terminal connected to that line:

```
stty
```

If it is impossible to log in at that terminal, you can use another terminal to display the characteristics. Log in as the super user at another terminal, and enter:

stty < *ttyname*

where ***ttyname*** is the name of the device special file corresponding to the serial line (see the "UNIX directories and special device files" chapter in this guide). For example, this command displays the current characteristics of the serial line named */dev/tty1a*:

```
stty < /dev/tty1a
```

The command displays the baud rate, the parity scheme, and other information about the serial line. This information is explained in the **stty**(C) manual pages.

One common change to a serial line is changing the baud rate. This is usually done from a terminal connected to another serial line because changing the rate disrupts communication between the terminal and the system. Before you can change the rate, you need to know the current baud rate of the terminal (review the terminal hardware manual to see how to determine the current baud rate). Once you have the baud rate, log in as the super user at the other terminal, and enter:

(stty *baud-rate* ;while : ; do sleep 3600; done)< *ttyname* &

where ***baud-rate*** is the current baud rate of the terminal, and ***ttyname*** is the name of the device special file corresponding to the serial line you wish to change. The baud rate must be in the set 50, 75, 110, 134, 150, 200, 300, 600, 1200, 2400, 4800, and 9600. For example:

```
(stty 9600 ;while : ; do sleep 3600; done)< /dev/tty1a &
```

This command changes the baud rate of the serial line */dev/tty1a* to 9600. Note that the "less than" symbol (<) is used for both displaying and setting the serial line from another terminal.

Another common change is the way the system processes input and output through the serial line. Such changes are usually made from the terminal connected to the serial line. For example:

```
stty tabs
```

This command causes the system to expand tabs with spaces (used with terminals that do not expand tabs on their own). Another example is:

```
stty echoe
```

This command causes the system to remove a deleted character from the terminal screen when you back over it with the ⟨Bksp⟩ key.

Note that the **stty** command may also be used to adapt a serial line to an unusual terminal, to another type of serial device that requires parity generation and detection, or to special input and output processing.

For a full description of this command, see the **stty**(C) manual pages.

Setting the terminal type

UNIX systems require that the terminal type be clearly defined before any work is done at the terminal. The preferable method for setting your terminal type is to assign the type to the **TERM** variable, a special environment variable that associates the terminal you are using with a list of characteristics given in the */etc/termcap* file. The characteristics tell the system how to interpret your terminal's keys and how to display data on your terminal screen.

If you are using the Bourne or Korn shell (**sh**(C) or **ksh**(C)), the **TERM** assignment has the form:

TERM=*termtype*; export TERM

If you are using the C shell (**csh**(C)), the **TERM** assignment has the form:

setenv TERM *termtype*

The ***termtype*** must be one of the names associated with one of the terminals defined in the */etc/termcap* file. The assignment must be entered at the terminal whose type you are setting.

For example, to set the terminal type to "ansi" from Bourne shell, go to the terminal you wish to set, enter at the shell prompt " $ ":

```
TERM=ansi; export TERM
```

Press the ⟨Return⟩ key. From the C shell, enter at the shell prompt " % ":

```
setenv TERM ansi
```

Press ⟨Return⟩.

If you are not sure which name you may use for ***termtype***, you can view the names either by displaying the */etc/termcap* file, or by reading the **terminals**(M) manual page which lists all terminals supported in the */etc/termcap* file. To display the file itself, enter:

more /etc/termcap

Press the ⟨Return⟩ key.

You can let the system define the terminal type automatically whenever you log in by including the **TERM** assignment in your *.profile* file (see "Modifying .profile and .login files" in the "Customizing system startup" appendix of this guide).

For an alternate method of setting your terminal type, see the manual page for **tset**(C).

If you let the system set the terminal type, be careful when logging in on terminals that are not the same as your normal terminal. The system has no way of checking whether or not the terminal assignment is correct for the given terminal and assumes that it is the same as your normal terminal. If it is not, you must set the terminal type manually.

Setting the terminal type automatically

If you want to have the terminal type set automatically at login time, follow this procedure:

1. Log in on the terminal in question and determine which ***ttyname*** you are using by entering the **tty**(C) command:

   tty

2. Log in as *root* and edit the file */etc/ttytype* with a text editor. Change the terminal type field for the line associated with the terminal in question to the terminal type you desire to use. Follow the model for the console. If you want your terminal type to be set to "wy50" for */dev/tty1a*, edit */etc/ttytype* as follows:

   ```
   wy50 tty1a
   ```

3. Then the user's start up file must be edited with the appropriate **tset**(C) command line to set the terminal type automatically. In each C-shell user's *.login* file, add the following line:

   ```
   tset -s -Q > /tmp/tset$$; source /tmp/tset$$; /bin/rm /tmp/tset$$
   ```

   Be sure to remove the default **setenv**(C) command line involving **TERM** and **TERMCAP** from the *.login* file.

In each Bourne shell or Korn shell user's *.profile*, add the following line:

```
eval 'tset -s'
```

Be sure to remove the existing **tset** command line from the *.profile* file.

4. Have all users log out, then log in again to test the new terminal type change. After they log in, have them verify the new term type by entering the **env**(C) command:

   env

Removing a terminal

From time to time it may be necessary to remove a terminal from the system, for example, if you wish to replace it with some other device. Before you can remove a terminal, you must disable it with the **disable**(C) command.

To remove a terminal, follow these steps:

1. Turn off the power to the terminal.
2. Log in as the super user at another terminal.
3. Use the **disable** command to disable the terminal. The command has the form:

   disable ***ttyname***

   where ***ttyname*** is the name of the serial line to which the terminal is attached. For example:

   disable tty1a

   This command disables the terminal connected to serial line */dev/tty1a*.
4. Disconnect the terminal from the system.

The serial line previously connected to the terminal is now free to accept another device.

Setting up a serial console

You can configure a serial device, rather than a display adapter, as your system console. The **boot**(HW) program sets the default console at boot time according to the following procedure:

1. The **boot** program looks for the entry SYSTTY=*x* (where *x* is a number that specifies the system console device) in the */etc/default/boot* file.
2. If the SYSTTY entry is not found or the */etc/default/boot* file is not readable, **boot** checks your system for a display adapter and designates it as your system console.

3. If no display adapter is found, **boot** looks for tty1a, sets the serial port to 9600 baud, 8 data bits, 1 stop bit, and no parity, and uses it as the system console.

To set up a serial console, create the following entry in your */etc/default/boot* file (where *x* is "0" for a display adapter or "1" for a COM1 serial port):

```
SYSTTY=x
```

To change the system console device from the command line, enter **systty=*x*** at the boot prompt (where *x* is "cn" for a display adapter or "sio" for a COM1 serial port). This does not create or change a **SYSTTY** entry in the */etc/default/boot* file.

Using serial multiscreens with mscreen

If you are familiar with **multiscreen**(M), the feature that provides many separate login screens on the console, it is possible to use a similar feature on a terminal. Terminals that have multiple pages of screen memory can be used as separate screens, each with a different login session, as if you had several terminals at your service instead of one.

On a Wyse 60 terminal, the contents of two entire screens of activity can easily be saved. The use of a third screen on the Wyse 60 is discussed below. This means that using two screens is very much like having more than one terminal. The complete functionality of a login session is provided on each screen, and previously executed commands (or their results) are displayed on each screen when it is in use. This section focuses on the Wyse 60, using its two pages of screen memory as the basis for all examples. (See the **mscreen**(M) manual page for a technical explanation).

You can also limit the number of mscreens available. The **mscreen** utility provides access to multiple terminal sessions, much like logging in on more than one terminal. These sessions are provided on "pseudo-ttys" rather than the tty devices usually used by terminals or modems. A tty is a special file associated directly with a particular hardware device used for communication with equipment such as terminals or printers. ttys can be seen in the */dev* directory as files with the name *tty* followed by a number and a letter.

A pseudo-tty is a device that is not associated with any real hardware, and it is used to simulate the function of a real tty. Users of networking products should already be familiar with pseudo-ttys, as they are the devices used to log in on remote machines. A pseudo-tty is represented by two software devices that appear in a listing of */dev* as "ptyp" and "ttyp", each followed by a number. The former is called the "master" tty and the latter the "slave". Between the two, they simulate a functional tty.

As installed, the system does not have any pseudo-ttys. Pseudo-ttys are created with the **mkdev ptty** command.

To configure the pseudo-ttys, log in as *root* and enter the following command:

```
mkdev ptty
```

This automatically creates the necessary devices and updates the files */etc/inittab* and */etc/conf/cf.d/init.base*, and updates the **NSPTTYS** kernel parameter (the maximum number of pseudo-ttys) as necessary. If this value is increased, a kernel relink will be necessary.

Adding more **mscreen** capability to your system should increase the productivity of the users. However, too much of a good thing can slow your system down. A system with 10 users, all of whom use two screens, could make your system perform as though it is servicing 20 users. Keep system performance in mind when deciding how many mscreens should be allowed system-wide, and who should be able to use them.

NOTE When using the **who** command, each user **mscreen** session is listed. If you wish to list only the master logins, use the **who -f** command.

No terminal known contains enough screen memory to save the material displayed during the use of all 20 logins that **mscreen** is capable of. However, any terminal should allow the user to switch between as many as 20 screens, providing the keyboard has enough extra keys to indicate the switch between screens. Note that the user will probably not find multiple screens very useful without multiple pages of screen memory. It is inconvenient, for example, to have to redraw the terminal's screen each time one switches screens when using a spreadsheet on one screen, and **vi** on the other. Most people who use terminals with minimal screen memory prefer shell layers **shl**(C) to **mscreen** for multiple login sessions. For more information, see the **shl**(C) manual page.

Troubleshooting

Unlike many utilities, **mscreen**'s complex responsibilities require a number of conditions for correct functionality. By following the suggestions here, you should be able to avoid some of the more common mistakes made by new **mscreen** users.

In preparing to use **mscreen**, make sure your terminal works with the program. Find out how much screen memory is provided by consulting your terminal manual. The **mscreen** utility uses the file */etc/mscreencap* to determine how to change screen images for your particular terminal. As shipped, */etc/mscreencap* is supplied with only a few terminals. This is not to say that other terminals do not work with **mscreen**; they do. You need only configure the */etc/mscreencap* file before using your terminal. If you run **mscreen** on a terminal that does not have an entry in */etc/mscreencap*, **mscreen** fails.

If you are sure your terminal works with **mscreen**, and you have a working *mscreencap*, but **mscreen** still fails, check the following common problems:

- Create more pseudo-ttys with **mkdev ptty**.

  You may need to create more pseudo-ttys if the pseudo-ttys currently on your system are in use.

- Verify switching.

  Make sure the */etc/mscreencap* for your terminal is correct. Use one of the examples in */etc/mscreencap* to check the way your function key output sequence is mapped to a particular **mscreen** command. You must log in separately to each screen you intend to use.

- Kill **mscreen** processes.

  If you are testing an *mscreencap* entry and you have trouble with the screens, you should do the following:

  1. Check the processes that are running:

     ps -u*username*

  2. Kill all the **mscreen** processes:

     kill -9 *process_numbers*

Advanced uses

Many users find **mscreen** satisfactory as provided. For advanced **mscreen** users, or anyone interested in learning more about both **mscreen** and the operating system, here are some "tuning" tips for using and extending **mscreen.**

In addition to invoking **mscreen** automatically, the script in Example 14-1 allows three full-featured mscreens on a Wyse 60 and adds a number of convenience features for the **mscreen** user. Example 14-1 presents the same material for the Bourne and Korn shell *.profile* file. Note that these examples are designed to be added to the end of your *.login* or *.profile* file, and replace any existing **tset** material.

Example 14-1 .login script

```
#
# Example material for the end of a C-Shell .login file.
#
# If logging in via pseudo-tty, suppress terminal initialization.
set ttyname=`tty`

# Set init to null, initially.
set init = ""

set noglob

# Reset init to the value "-I" when logging in on a pseudo-tty to
# suppress the tset terminal initializations string.

if ( `expr $ttyname : "/dev/ttyp"` > 0 ) set init = "-I"
set term = (`tset -m ansi:ansi -m wy60:wy60 -m:\?wy60 -r -S -Q $init`)
setenv TERM $term
unset noglob term
# Put WYSE 60 in ECON-80 mode during initial log in process.
if ( "$init" != "-I" && "$TERM" == "wy60" ) /bin/echo "\033eG\c"

# Set the prompt to indicate the tty number of the current
# mscreen and command.
set prompt = "`expr $ttyname : '/dev/\(.*\)'` \!% "

# Release the local variables used.
unset ttyname init

# Run mscreen and logout if the 'stop' key (defined as S-F9 in
# the default /etc/mscreencap for wy60) is pressed. This string
# is described in the mscreen (M) manual pages.
mscreen -n 3
if($status == 0) logout
```

Example 14-2 .profile script

```
#
# Example material for the end of a Bourne shell .profile file.
#
ttyname=`tty`

init=""

if [ `expr $ttyname : "/dev/ttyp"` -gt "0" ]
then
        init="-I"
fi
eval `tset -m ansi:ansi -m wy60:wy60 -m :\?wy60 -r -s -Q $init`
export PATH

if [ "$init" = "-I" -a "$TERM" = "wy60" ]
then
        /bin/echo "\033eG\c"
fi

PS1="`expr $ttyname : '/dev/\(.*\)'` $ "

unset ttyname init

mscreen -n 3
if [ "$?" = "0" ]
then
        exit
fi
```

Many *termcap* entries (including wy60) clear the screen buffers (that **mscreen** uses to store the contents of multiple screens) as part of the initialization string. In Examples 14-1 and 14-2, **tset**(C) sends the initialization string only during the first login procedure. When logging in on pseudo-ttys, **tset** is invoked with the **-I** flag. This is done by adding the **init** variable to the **tset** line. The first time **tset** is run, **init** has a value equal to "", adding nothing to the **tset** command. When it is run subsequently, **init** has a value of "-I", adding the option to **tset**.

Following the **tset** command, during first login procedure, the string "\033eG\c" is echoed. This escape sequence changes the **COLUMNS** setting in the Wyse 60 to ECON-80 mode. The combination of these settings frees up just enough screen memory to use three screens. As an extra convenience, the user's prompt is set to display the current slave pseudo-tty number, allowing the user to keep track of which screen is in use.

If you do not use a Wyse 60 terminal, you can still set your prompt to indicate the current screen, and invoke **mscreen** automatically while checking for the shell return code, as illustrated in Examples 14-1 and 14-2.

Using scancode-compatible terminals

Most terminals send information to the operating system only in the form of keytop values, which are the characters that appear on the faces of the keys. However, a few terminals can also send PC scancodes, which are unique values associated with the depression and the release of each key. Several applications and environments now use PC scancodes and more are under development.

A scancode application running on a terminal that is in PC-scancode mode can access more distinct keystrokes than character mode would provide. For example, if you set your terminal to character mode and press the key labeled "A", your terminal sends a single value (the ASCII value of " a ") to your application. However, if you set your terminal to scancode mode and press the key labeled "A", your terminal sends one value when you depress the key and a second value when you release the key. A scancode application translates these scancode values according to a predetermined map.

Setting up a scancode-compatible terminal

Your scancode-compatible terminal can reside in either character mode or scancode mode. If you choose to leave your terminal in character mode, then each time you start an application that uses scancodes (for example, MS Word), the application switches the terminal to PC-scancode mode. When you quit the application, it returns the terminal to character mode. The screen flashes each time the terminal mode changes, and the switch adds a few seconds delay to starting and quitting your scancode application. For these and other reasons, we recommend that you run a terminal in scancode mode at all times, instead of letting the scancode application switch terminal modes.

NOTE You might encounter problems using PC-scancode mode with a smart serial card. For example, scancode mode might interfere with XON/XOFF flow control. If this happens, consult your card's documentation for the manufacturer's recommendations on resetting the flow control start and stop characters. If your card does not support changing the start and stop characters, or if you experiençe a problem unrelated to flow control, consult your card's documentation to determine whether you can reset the card so that it no longer takes on line-discipline processing for your scancode lines.

The following two subsections describe how to configure your system when you run a terminal in scancode mode at all times. If you choose to leave your terminals in character mode, you do not need to configure your system specially to use a scancode-compatible terminal.

Setting up scancode mode for all sessions

If you want a terminal to reside in scancode mode, you need to modify certain files. To determine which files you need to modify, enter the following command as super user, where ***ttyline*** is the tty number for your scancode terminal:

disable /dev/*ttyline*

You see messages naming two initialization files associated with that tty. Write down the filenames. One file is */etc/inittab* and the other is either */etc/conf/cf.d/init.base* or a file from the */etc/conf/init.d* directory. Edit the files that the screen displays, and, on the line that corresponds to the correct tty, change the last field from "m" to "sc_m". For example, if you want to run tty001 in scancode mode, change the line:

```
001:2:off:etc/getty tty001 m
```

to read:

```
001:2:off:etc/getty tty001 sc_m
```

For more information on modifying these initialization files, see the section on adding a terminal earlier in this chapter.

NOTE For each Wyse 60 or Wyse 150 terminal that you want to run in scancode mode, change the user's environment by specifying the "wy60-pc" **TERM** environment variable in the user's *.login* or *.profile* file or in */etc/ttytype*. (The Wyse 150 behaves the same in scancode mode as the Wyse 60.) For instructions on defining the **TERM** variable, see the section on setting terminal type earlier in this chapter.

Note that only Wyse terminals have the "-pc" names, others, for example, hp700, are the same in scancode and ASCII mode.

After you edit the two initialization files, set the terminal itself to scancode mode (some manufacturers refer to "PC-personality"). Consult your terminal documentation for instructions on setting this mode.

Finally, enter the following command to reenable the terminal line:

enable /dev/*ttyline*

Setting up scancode mode for one session

If you are unsure whether you want to run a terminal in scancode mode when you are not using a scancode application, you can experiment by using scancode mode for a single session. Use the **scanon** command to set your terminal and your line discipline to scancode mode. The **scanon**(M) manual page describes the **scanon** and **scanoff** commands.

Using function keys in scancode mode

When you set up your terminal and system to run in scancode mode, your function keys get set to their default values. If you want to program your function keys while you work in scancode mode, you must use the **setkey**(C) or **mapstr**(M) utility, rather than your terminal's setup procedure. The **setkey** command lets you program one key at a time, while **mapstr -f** reads a file containing the assignment for all the function keys. These utilities formerly affected only the console.

NOTE **scanon** does not run **mapstr**, so if you use **scanon** you also have to run **mapstr** to use the fkeys, numeric pad and arrow keys.

The syntax for the **setkey** command is:

setkey ***keynum string***

The **setkey** command assigns the specified ANSI *string* to be the output of the function key ***keynum***. For example, for function key 1 (⟨F1⟩) to output the string "date", use this command:

setkey 1 "date"

For a key assignment to last beyond the current login session, place the **setkey** command in your *.login* file.

The syntax for the **mapstr** command is:

mapstr [-d] [*datafile*] [-f] [*termtype*]

Without the **-d** option, **mapstr -f** reads the function key values from the file in */usr/lib/keyboard/strings.d* that corresponds to the terminal type. To customize your function key assignments, create a new file for **mapstr** to read, using a file from */usr/lib/keyboard/strings.d* as a template. Then specify your new file in the **mapstr** command as follows:

mapstr -d *newfile* -f

For these key assignments to last beyond the current login session, place the **mapstr** command in your *.login* file.

Correcting a hung scancode-compatible terminal

If your PC-scancode application crashes, your terminal might hang with the terminal and the line discipline in incompatible modes. To correct this incompatibility, log into another terminal and use either the **scanon**(M) or **scanoff**(M) command as described below.

If you want to restore your terminal and line discipline to PC-scancode mode, enter the following **scanon** command, where ***ttyline*** is the tty of the hung terminal:

```
scanon /dev/ttyline
```

If you want to restore your terminal and line discipline to character mode, use the **scanoff** command:

```
scanoff /dev/ttyline
```

You do not need to be super user to use **scanon** and **scanoff** to affect your own tty. For more information on **scanon** and **scanoff**, see the **scanon**(M) manual page.

Chapter 15 Using modems

Modems (from *mo*dulate *dem*odulate) are a significant addition to your system, allowing you to communicate over phone lines from remote sites.

This chapter explains the following tasks:

- connecting modems to your computer
- maintaining modem connections
- defining passwords for dial-in lines

Note that physical connections between a device and the system vary according to hardware configuration. For specific information about connecting your serial device, refer to the hardware manuals provided with the device and with your computer. Before adding a modem, you should make certain you have a port available, either directly on COM1 or COM2, or from a multiport card. Configuring serial ports is discussed in the "Adding multiport cards, memory, and other bus cards" chapter of this guide.

Choosing a serial port

The system supports modem control on serial ports. Table 15.1 contains sample device names of serial ports with and without modem control.

Table 15-1 Serial ports

Device	Function
/dev/tty1a	main serial adapter without modem control
/dev/tty1A	main serial adapter with modem control
/dev/tty2a	alternate serial adapter without modem control
/dev/tty2A	alternate serial adapter with modem control

/dev/tty1a and */dev/tty1A* refer to the same serial port (likewise for */dev/tty2a* and */dev/tty2A*). The operating system uses different device-driver subroutines for each. Never attempt to use both modem and non-modem control ports at the same time or you will see the warning:

```
cannot open: device busy
```

For systems including multiport serial cards, */dev/tty*[*1,2*][*a-m*] are the non-modem control devices, and */dev/tty*[*1,2*][*A-M*] are the modem control devices.

Configuring your modem

Proper modem configuration is necessary when using **cu**(C) and **uucp**(C). Modem settings differ for each modem. Consult your modem manual for the proper switch settings.

Smartmodem 1200 or compatible

If you have a Hayes Smartmodem 1200 or compatible, switches 3 and 8 should be down:

	1	2	3	4	5	6	7	8
up	•	•		•	•	•	•	
down			•					•

When switch 3 is down, the resulting codes are sent by (echoed by) the modem to the terminal or computer. When switch 8 is down, the modem can interpret the command being issued. This allows both MS-DOS and UNIX system communications systems to work.

Table 15-2 lists the functions of each switch setting.

Table 15-2 Hayes-Compatible switch settings

Switch	Position	Function
1	up*	Modem responds to DTR from computer
	down	Modem forces DTR high, so no signal is required from computer
2	up*	Result codes are in English
	down	Result codes are numeric
3	up	There are no result codes
	down*	Result codes are sent in response to each modem command
4	up*	Commands are echoed
	down	Commands are not echoed
5	up*	Modem answers phone
	down	Modem does not answer phone
6	up*	CD is asserted when carrier is actually present
	down	CD and DSR are forced high
7	up*	Modem is attached to single-line phone
	down	Modem is attached to multi-line phone
8	up	Modem does not recognize dialing commands
	down*	Modem recognizes dialing commands

The asterisks indicate the switch settings required. If you have a different modem, consult your reference manual for the proper switch settings to both send and receive calls.

Hayes 2400 and V-series 9600 Smartmodems or compatible

The Hayes 2400 and V-series 9600 Smartmodems or compatible modems are supported and are configured automatically.

Telebit Trailblazer

If you have a Telebit Trailblazer modem or compatible, log in as *root* and enter the following command:

```
/usr/lib/uucp/dialTBIT -z /dev/ttynn 9600
```

where *nn* is the tty number of the serial line.

General modem configuration guidelines

When configuring non-supported modems, there are a number of things you should look out for:

- Do not enable any compression mechanism when using the modem for UUCP. These compression mechanisms are only really effective for textual data, producing much smaller gains for binary data, and actually increasing the data size for already compressed data.
- Do not enable MNP or any other form of error correction/ detection when using the modem for UUCP. UUCP has its own error detection scheme, and this conflicts with any lower level scheme, giving rise to poorer throughput and possible failed transfers.
- Do not enable XON/XOFF flow control when using the modem for UUCP. The XON and XOFF characters can get interpreted as part of the UUCP protocol data stream, and thus cause retransmissions and transfer failures. Make sure that if XON/XOFF flow control is turned off, any XON/XOFF passthrough mode that your modem has is turned on.

  You can use RTS/CTS flow control with your modem, but you will need to write a dialer program to use it. An example of a dialer program that uses RTS/CTS is the Trailblazer dialer. You can find the source for this in */usr/lib/uucp/dialTBIT.c.*

If you want to enable any of these features when using the modem for **cu**(C), then you will need to have an alternate dialer defined. This can be done by modifying and renaming the Dialers file entry, or by modifying and renaming the dialer program, such that you have one dialer definition for UUCP and another for **cu**.

Installing a modem

NOTE We strongly advise that you do not use an internal modem. We have successfully tested the Hayes 1200 and 2400 baud internal modems, and these modems are known to work. However, most Hayes-compatible internal modems have compatibility problems which prevent them from working properly under UNIX System V. These incompatibilities include:

- No support for "quiet mode," "no-echo mode," or auto-answer mode.
- Modem floods the bus with spurious interrupts.

When you are hooking up your modem, or any other device, make sure that serial wires connected to your computer are not left hanging. An unterminated line connected to your computer can considerably reduce system performance; always unplug a modem wire at the computer end instead of at the modem end.

Three-wire cables often used to connect terminals to the computer are not sufficient for connecting modems. For a modem cable on a 25-pin serial port, pins 2, 3, 7, 8, and 20 must be connected straight through. If you are unsure as to what to use, a cable that connects all pins works correctly. Either a ribbon cable, or what is called a "straight-through" cable, meaning that it connects the pins straight across, works. See **serial**(HW) for details of 9-pin connections.

To install your modem for dial-in or dial-out, follow these steps:

1. Make sure the UUCP package is installed. Use **custom**(ADM) to install it if necessary.
2. Make sure the serial port you have chosen for your modem is recognized at bootup (check */usr/adm/messages* or use **hwconfig**(C)) and, if the modem is internal, make sure that the COM port the internal modem is configured for does not conflict with any other device. Only serial devices attached to COM1 and COM2 are supported.
3. Make sure the modem and non-modem control ports are disabled by using the **disable**(C) command. For example, the following command applies to a modem connected to COM1:

   ```
   disable tty1a
   disable tty1A
   ```

4. If you are going to use the line for dial-out, the serial port must be owned by *uucp*. To make sure the line is owned by *uucp* enter this command:

   ```
   chown uucp /dev/ttynn
   chgrp uucp /dev/ttynn
   ```

   where *nn* is the number of the corresponding line.

5. Connect the modem to the machine using a straight-through cable (pins 2 and 3 are not crossed). The cable must have at least pins 2, 3, 7, 8, and 20 connected. (This is for a 25-pin serial port. See **serial**(HW) for 9-pin connections.)

   Most standard COM ports use straight-through cables, but some hardware requires a null-modem cable (pins 2 and 3 are crossed). A standard COM port is known as DTE (Data Terminal Equipment); a port that needs a null-modem cable is known as DCE (Data Communications Equipment). Check your hardware documentation if you are unsure. If the COM board is a DCE, you need a null-modem cable.

6. If you intend to use the modem for dial-in, make the entry for this serial port in the */etc/inittab* file look like the following:

   ```
   t1a:2:off:/etc/getty -t60 tty1a 2
   ```

   You can change an entry with a text editor. The last field of the */etc/inittab* entry is the line mode, which indicates the */etc/gettydefs* entry to use for this line. Entry **2** selects the 1200-2400-300 cycle. Use the **m** entry for 9600 baud. For more information on the */etc/inittab* file and the various control codes, see the **getty**(M) and **inittab**(F) manual pages.

   If you intend to use the modem for dial-out, check that the entry for this serial port in the */etc/inittab* file looks like the following:

   ```
   t1A:2:off:/etc/getty -t60 tty1A 3
   ```

   In this case the */etc/gettydefs* entry is **3**, which is the 2400 baud entry.

   If the line is to be shared between dial-in and dial-out, ensure that it has an appropriate entry in */usr/lib/uucp/Devices* and in */etc/inittab*.

   WARNING If you make any changes to */etc/inittab* by hand that you wish to be permanent, you must also make the same change to */etc/conf/init.d/sio*. This is because each time the kernel is relinked (as when a driver is added or a tunable parameter is changed), */etc/inittab* is reconstructed from the entries found in */etc/conf/init.d/sio*.

7. Add the correct entries to the */usr/lib/uucp/Devices* file. This file should have two entries for each serial port being used for a modem. One of the entries is used when you start a call using the modem (the Automatic Calling Unit (ACU) line), and the other line is used to configure the modem using the standard Hayes command set (the Direct line). For a Hayes-compatible modem operating at 2400 baud, using COM1, the entries in */usr/lib/uucp/Devices* should be:

   ```
   ACU  tty1A - 1200-2400 /usr/lib/uucp/dialHA24
   Direct  tty1a - 1200-2400 direct
   ```

   Make sure that the entries do not have a number sign " #" in front of them. This is the syntax to show that the line is only a comment, and is to be ignored. There are many examples in the *Devices* file that are commented out with this character.

8. Test the modem's ability to dial correctly. (This process is outlined in Figure 15-1.) For example, the following command uses *tty1a*:

   cu -ltty1a dir

   You should see a message indicating that you are connected. If you see the message ``cu: dir permission denied,'' the user executing the **cu** command does not have write permission on the */usr/lib/uucp/Devices* file.

   If you do not see such a message, and there was no message to indicate that you connected correctly, then the **cu** command is incorrect, the *Devices* file is incorrect, or the serial port is not operating correctly.

   NOTE The instructions that follow assume a Hayes-compatible command set and response codes. Other modems may use other conventions. Consult your modem documentation for further details.

9. If you see a message confirming your connection, enter the following on your keyboard:

   AT

   The screen should echo "OK". If the modem is set to return result codes as numeric codes rather than text, you see "0".

10. If this does not occur, check that the "receive" light on the modem flashes when you press a key. This indicates the modem is receiving signals from the keyboard. If this light is not flashing, check your cable and modem switch settings. If the "receive" light flashes, but you still do not get an "OK" response from the modem, enable the modem's echo capability and responses to commands by entering:

    ATE1Q0

11. If your terminal still does not display "OK" or "0", connect the modem to another port (COM1 or COM2). If the modem works with the new port, check that the device is using the correct interrupt vector. (Serial port COM1 uses interrupt **4**; COM2 uses interrupt **3**.)

12. If the terminal does not echo the "OK" message when it is connected to the new port, the modem is defective; see your hardware documentation.

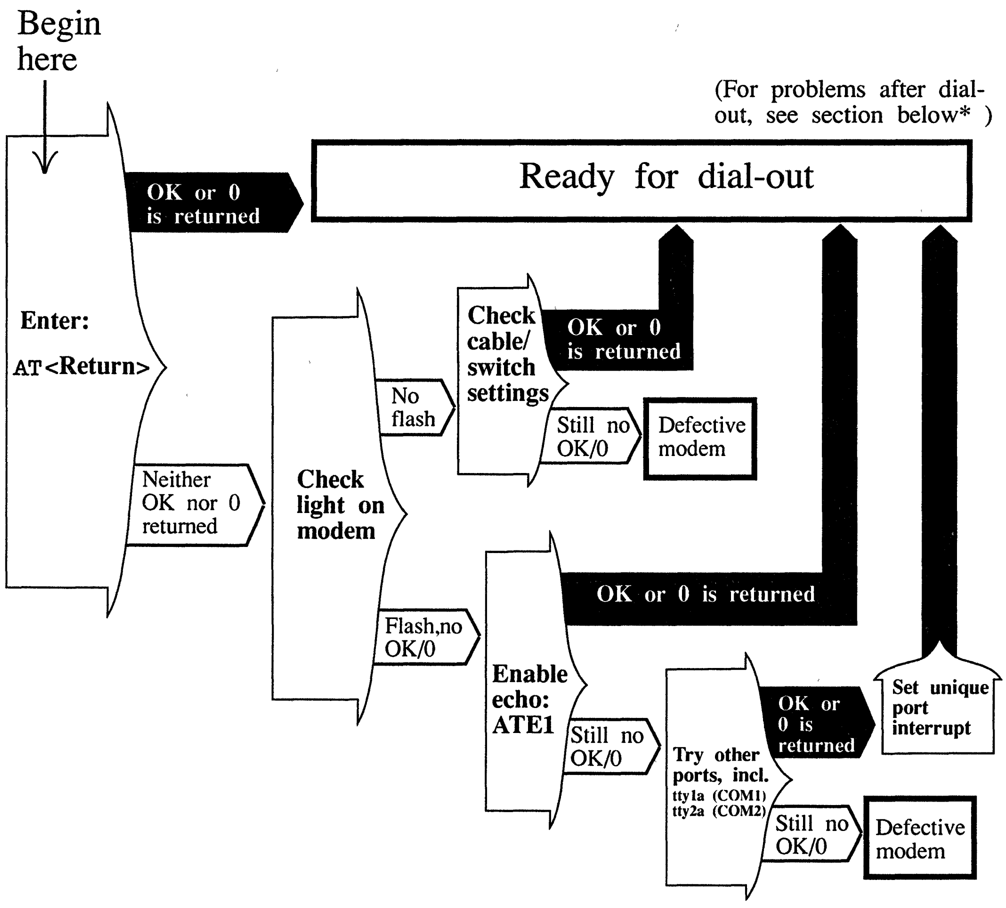


Figure 15-1 Testing your modem connection

13. If you plan to use the modem primarily for dial-in, skip to step 15. To test the modem for dial-out, enter:

ATDT*phonenumber*	for tone dial phone connection
ATDP*phonenumber*	for pulse dial phone connection

When you confirm that the modem can dial out, exit **cu** by entering:

```
~.
```

Then press ⟨Return⟩.

You are now ready to dial into another system. Use the following command to dial out:

```
cu -ltty1A 5551212
```

Change "5551212" to the phone number of the system into which you are dialing.

NOTE When including a phone number in the **cu** command to connect with a modem, you should not use the comma " , " from the Hayes command set to indicate a pause. Use hyphens instead. This also applies to *Systems* file entries.

If you have any problems, refer to the section on troubleshooting your dial-out modem in the "Troubleshooting your system" chapter in this guide.

14. If the line is also to be used for dial-in, you must follow some additional steps to configure the modem. Some modems have switches or software commands for setting the modem configuration. If your modem has such settings, configure it as instructed in your modem manual. The modem should be configured as follows:

    - The modem must be set to Auto-answer. Your modem must support auto-answer; most internal modems do not have auto-answer and some external modems do not have this setting.
    - It should not answer when the DTR (Data Terminal Ready) line is not active, and should disconnect from the current connection when DTR goes from active to inactive.
    - The CD (Carrier Detect) line should follow the incoming carrier; it should go low when a carrier is present, high when a carrier is not present.
    - Echo should be turned off; it should not echo characters sent to it.
    - It should be set for quiet mode, sending response codes instead of response strings.

15. Enable the port you are using for your modem with the **enable**(C) command:

    enable ***ttyname***

    where ***ttyname*** is the modem control port.

16. Dial this modem from another modem.

If you have any problems, refer to the section on troubleshooting your dial-in modem in the "Troubleshooting your system" chapter in this guide.

Dialing into your computer

To allow dialing into your computer, you must enable a serial line that recognizes modem control signals with the **enable**(C) command.

To use the main serial adapter (COM1), enter:

disable tty1a
enable tty1A

For the alternate serial adapter (COM2), enter:

disable tty2a
enable tty2A

Note that tty1A and tty1a refer to the same (main) serial line, and tty2A and tty2a refer to the same (alternate) serial line. Do not enable the same line in both its modem and non-modem modes at the same time, as this causes an error.

Adding passwords for dial-in lines

If desired, you can define special dial-in passwords on selected tty lines, requiring selected classes of users to input dial-in passwords. Logging information, including the last time of connection, can be stored for later use.

Specific dial-in lines that require passwords are defined in the file */etc/dialups*. The format is one tty device name per line, for example:

```
/dev/tty1A
/dev/tty5C
```

The actual dialup passwords are kept in the file */etc/d_passwd*. The password format is the same one used in */etc/passwd*. The first field ("user name") in */etc/d_passwd* is not a user name, but the name of a shell program (for example, */bin/sh*) used in */etc/passwd*. If the login shell of the user attempting to log in (on a tty line listed in */etc/dialups*) is listed in */etc/d_passwd*, then the user is prompted for the dial-in password stored in */etc/d_passwd*.

Here is the syntax for creating a dial-in password:

passwd -m *dialname*

Change the password for dialup shell ***dialname*** (listed in */etc/d_passwd*). If ***dialname*** begins with a slash " / " the entire shell name must match. Otherwise the password for every shell whose basename is ***dialname*** is changed. Only the super user can change a dialup shell password.

Dialing out from your computer

The **cu**(C) and **uucp**(C) utilities call remote systems and transfer data on UNIX systems. The file */usr/lib/uucp/Devices* (referred to as *Devices*) contains information used by these programs to determine the characteristics of a particular serial line.

The *Devices* file contains lines that specify the device for the line, the call-unit associated with the line, and the baud rate, which are to be used by UUCP. (Modem control devices should be used with lines connected to modems.)

Using dialer programs

For dialing, both **cu** and UUCP use a common set of dialers, which can be standalone binaries (programs) like */usr/lib/uucp/dialHA12*, or entries from the file */usr/lib/uucp/Dialers*.

NOTE For best results, dialer programs are preferred over *Dialers* entries. The following *Devices* entry is an example using a dialer binary:

```
ACU  ttynn  -  300-2400  /usr/lib/uucp/dialHA24
```

Dialer binaries

Table 15.3 lists the dialer binaries provided with your distribution. They are supplied in binary and source form.

Table 15-3 Dialer binaries

Binary file	Modem
dialHA12	Hayes Smartmodem 1200 or compatible
dialHA24	Hayes Smartmodem 2400 or compatible
dialHA96V	Hayes Smartmodem 9600 or compatible
dialMUL	Multitech Multimodem 224 EH
dialVA3450	Racal Vadic 3451 modem
dialT1500	Trailblazer TB1500
dialTBIT	Telebit Trailblazer Modem

The source and a make file for recompiling are included in the directory */usr/lib/uucp*. If you have any other kind of modem, you can modify any of the source files and create your own dialer program. Note that you must have the Development System installed to compile a program.

To make a new dial program, follow these steps:

1. Change directory to */usr/lib/uucp* with the following command:

   cd /usr/lib/uucp

2. Edit the file *makefile* in the directory */usr/lib/uucp* and find the line that reads:

   ```
   EXES= dialHA12 dialHA24 dialHA96V dialMUL dialTBIT dialT1500 dialVA3450
   ```

   Add the name of the dialer program that you wish to use. When this is done, exit the file, saving the changes you made.

3. Next, enter the following command at your shell prompt:

   make

   Then press ⟨Return⟩.

   When the **make** command is finished, you have a new dialer program. This can be used in the fifth field of an entry in the *Devices* field.

Dialers file entries

Table 15.4 lists dialer types that are available as *Dialers* entries.

Table 15-4 Dialers file entries

Dialer type	Modem or Data Switch
Direct	direct line; no dialer
Penril	Penril modem
Hayes	Hayes modem (or compatible)
Ventel	Ventel 212+ modem
Vadic	Racal Vadic 3451 modem
Vadic9600	Vadic 9600VP
LANswitch	network switch described in type field
Hayes1200	Hayes Smartmodem 1200
Hayes2400	Hayes Smartmodem 2400
Develcon	Develcon network dataswitch
Micom	Micom network dataswitch
Rixon	Rixon Intelligent Modem
ATT4000	AT&T Programmable 300/1200 Modem Model 4000
ATT2212c	AT&T DATAPHONE II 2212C Modem
ATT2224	AT&T DATAPHONE II 2224 Modem
NLS	Network Listener Service

The following is an example *Devices* entry that uses a *Dialers* file entry:

```
ACU tty1A - D1200 hayes1200
```

Making a dialer

The UNIX system provides software to operate high speed modem devices. A program is included that enables you to create dialers for high speed modems without the need for a development system. Default dialer entries are included for:

- Hayes Ultra 96-2000US
- MICC 9610
- MULTITECH MT932EA
- Telebit T2500
- U.S. Robotics Courier V.32bis

To create a dialer you should understand how the modem works. This section helps you to understand the registers and what they do in a particular modem.

NOTE Many different modems are available in the marketplace. They range in speed from 300 to 38400 bps. SCO does not support serial communication speeds greater than 9600 bps. In some cases, higher speeds are known to work. Some modems also have data compression while others do not.

The software allows you to enable or disable the desired options by defining the "setup strings." Thus, each modem can be configured by putting the appropriate commands into the strings to be issued to the modem.

Some newer, high speed modems can run the serial port at specified speeds, while the modem varies the line speed as needed. As discussed later, the modem must be prepared for this option by putting the appropriate commands in the "option string" and then ensuring that the UNIX system always "talks" to the modem at that speed. This preparation requires establishing */usr/lib/uucp/Devices* to have the set speed as the highest speed available and */etc/inittab* to use that same speed.

You must also tell the program **make.dialer** that the connect speed for all connections is **NOT USED,** except for the 9600 entry. That entry can be modified to CONNECT 2400/rel, indicating a 2400 bps connection with 4X data compression on the MICC MNP Class 5 modem. This step creates a 9600 connection between two machines and may require adjustment to the registers and serial line settings.

Each time **uugetty** is run on a modem control port, the first dialer associated with that port in */usr/lib/uucp/Devices* is executed to set up the modem for dial-in. This process is necessary to ensure that the appropriate dialer is listed for each modem in this file.

Although different dialers can run one modem if the default setup dialer is listed first, additional adjustments may be required.

atdialer

The key to the entire Configurable Dialer package, the program **atdialer,** is the program that "talks" to the modem. The source code included with this package details how the program functions; the following discussion provides an overview of that process.

atdialer reads information about the modem. This information enables **atdialer** to work, and includes data the program needs to talk to a particular modem. For example, **atdialer** must know the entry to set up a modem, and the response expected from the modem. If "AT" were typed to a Hayes Compatible modem, the response expected from the modem would be "OK". **atdialer** looks for this information in a simple text file, created using a program called **make.dialer. make.dialer** is the support program that allows the text file from which **atdialer** reads modem specific information to be created.

The concepts discussed below are useful for understanding how **atdialer** functions. With the UNIX system you can execute a program by typing its name on the command line. The name typed to run a program is called "argv[0]". You can execute that same program by typing in a different name if a new name is linked to the old name. For example, if a program called **TEST** can be executed by typing "TEST" at the command line, the name "TEST01" can be linked to **TEST.** You can then execute **TEST** by typing "TEST01" at the command line, known as the same program with a different "argv[0]". In this same manner, **atdialer** knows which information to read to control a modem. The program **atdialer** reads the information contained in */etc/default/argv[0].* Therefore, a file with the same name as the program executed must exist in the default directory and contain information about the modem. If an MICC modem is used, **atdialer** is linked to *atdialMICC* for execution and */etc/default/atdialMICC* contains the information needed to run the MICC modem.

In addition to the above functions, **atdialer** "talks" to the modem. After it is called from the command line, **cu, uucico,** or **uugetty, atdialer** puts the modem into dial-in mode or establishes a connection to another modem. It then returns a status to the calling program. The status is either an error or the speed at which the request was processed.

For example, if you type a **cu** command on the command line, **cu** calls the dialer program with arguments describing the speed, port number, and phone number. **atdialer** then attempts to make the modem on that port dial the given phone number and connect to a modem at the other end at the given speed. The modem then either makes the connection and returns a success status of the speed the connection was made, or it returns an error status, such as "BUSY" or "NO CARRIER".

make.dialer

The program **make.dialer** is the support file used to create data files for **atdialer** to read. Although you can use an editor to create the files that **atdialer** reads, this program provides a simple user interface, default values, and instructions for creating a new dialer.

This program creates a simple text file containing the information to run a particular modem. When **make.dialer** is executed, a short help screen is printed to the screen. This text emphasizes the need to understand the modem registers and suggests a naming convention for modem dialers. The program then asks for the name of the dialer to create and moves on to setup strings. Later, it will ask about answers the modem will return. This information is needed for **atdialer** to run properly. This program also gives instructions on how to build the new dialer by moving the text file to the appropriate location and linking another name to **atdialer**.

Make sure you create the proper */etc/gettydefs* entry for the modem. The modem must use RTS/CTS flow control if UUCP is selected and these restrictions also apply to both */etc/inittab* and */etc/conf/cf.d/init.base*. Also, ensure that */usr/lib/uucp/Devices, /usr/lib/uucp/Systems* and */usr/lib/uucp/Permissions* files are set up correctly.

Dialer creation example

1. Log in as root.
2. Change directories to */usr/lib/uucp*.

3. Enter the following command:

   ./make.dialer

   The following help message is displayed on your screen:

```
                          Some Help
                          ---------

The following string is an example of a modem setup string. It
demonstrates some standard things that need to be set up in order
to make a modem work with UNIX.
Please make sure that you understand what each of the options
does.
                  AT&F&D2&C1S0=1S2=043Q3&W

The following is an example of a good dialer name for brand XY modem

                          atdialXY

When you are entering strings into this program simply press q to go
on to the next question. You can hit the <del> key to quit out of the
program with no ill effects.

Press <Return> to continue:
```

   Press ⟨Return⟩ to continue, or ⟨Del⟩ to quit.

4. The following screen is displayed:

```
Please enter the name of the dialer to create: atdialXY
```

   Enter the name of the dialer to create and press ⟨Return⟩.

5. The following screen is displayed:

```
The current modem SETUP string is:  "AT&F&D2&C1S0=1S2=043\Q3&W"
Enter new modem SETUP string or 'q' to accept this one
-> q
```

   Enter **q** to accept the current setup string, or enter a different setup string and press ⟨Return⟩.

6. The following screen is displayed:

```
The current OPTION STRING is:

If you want to have a special register set in the modem when you
put an X or an x at the end of the phone number, then enter it
here. For example, one modem might like to see ATSP if it is to
do UUCP spoofing on a phone number with an x at the end.

Press q to continue without using this option

RE-Enter OPTION STRING or q to accept this one: q
```

   Enter **q** to accept the current option string, or enter a different option string and press ⟨Return⟩.

7. The following screen is displayed:

```
The current modem DIAL string is:  "ATDT"
Enter new modem DIAL string or 'q' to accept this one
-> q
```

   Enter **q** to accept the current modem dial string, or enter a different modem dial string and press ⟨Return⟩.

8. The following screen is displayed:

```
The current modem ESCAPE string is:  "+++"
Enter new modem ESCAPE string or 'q' to accept this one
-> q
```

   Enter **q** to accept the current modem escape string, or enter a different modem escape string and press ⟨Return⟩.

9. The following screen is displayed:

```
The current modem HANGUP string is:  "ATQ0H0"
Enter new modem HANGUP string or 'q' to accept this one
-> q
```

   Enter **q** to accept the current modem hangup string, or enter a different modem hangup string and press ⟨Return⟩.

10. The following screen is displayed:

```
The current modem RESET string is:  "ATQ0Z"
Enter new modem RESET string or 'q' to accept this one
-> q
```

    Enter **q** to accept the current modem reset string, or enter a different modem reset string and press ⟨Return⟩.

11. The following screen is displayed:

```
The current enable AUTOANSWER string is:  "ATS0=1Q1"
Enter new enable AUTOANSWER string or 'q' to accept this one
-> q
```

    Enter **q** to accept the current enable autoanswer string, or enter a different enable autoanswer string and press ⟨Return⟩.

12. The following screen is displayed:

```
The current get modem attention string is:  "AT"
Enter new get modem attention string or 'q' to accept this one
-> q
```

    Enter **q** to accept the current get modem attention string, or enter a different get modem attention string and press ⟨Return⟩.

13. The following screen is displayed:

```
The current modem DISABLE ESCAPE string is:  "ATS2=128"
Enter new modem DISABLE ESCAPE string or 'q' to accept this one
-> q
```

    Enter **q** to accept the current modem disable escape string, or enter a different modem disable escape string and press ⟨Return⟩.

14. The following screen is displayed:

```
The current result code for OK is:  "OK"
Enter new result code for OK or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for OK, or enter a different result code for OK and press ⟨Return⟩.

15. The following screen is displayed:

```
The current result code for NO CARRIER is:  "NO CARRIER"
Enter new result code for NO CARRIER or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for no carrier, or enter a different result code for no carrier and press ⟨Return⟩.

16. The following screen is displayed:

```
The current result code for ERROR is:  "ERROR"
Enter new result code for ERROR or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for error, or enter a different result code for error and press ⟨Return⟩.

17. The following screen is displayed:

```
The current result code for NO DIALTONE is:  "NO DIALTONE"
Enter new result code for NO DIALTONE or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for no dialtone, or enter a different result code for no dialtone and press ⟨Return⟩.

18. The following screen is displayed:

```
The current result code for BUSY is:  "BUSY"
Enter new result code for BUSY or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for busy, or enter a different result code for busy and press ⟨Return⟩.

19. The following screen is displayed:

```
The current result code for NO ANSWER is:  "NO ANSWER"
Enter new result code for NO ANSWER or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for no answer, or enter a different result code for no answer and press ⟨Return⟩.

20. The following screen is displayed:

```
The current result code for 300 baud connection is:  "CONNECT"
Enter new result code for 300 baud connection or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for 300 baud connection, or enter a new result code for 300 baud connection and press ⟨Return⟩.

21. The following screen is displayed:

```
The current result code for 1200 baud connection is:  "CONNECT 1200"
Enter new result code for 1200 baud connection or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for 1200 baud connection, or enter a different result code for 1200 baud connection and press ⟨Return⟩.

22. The following screen is displayed:

```
The current result code for 2400 baud connection is:  "CONNECT 2400"
Enter new result code for 2400 baud connection or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for 2400 baud connection, or enter a different result code for 2400 baud connection and press ⟨Return⟩.

23. The following screen is displayed:

```
The current result code for 4800 baud connection is:  "CONNECT 4800"
Enter new result code for 4800 baud connection or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for 4800 baud connection, or enter a different result code for 4800 baud connection and press ⟨Return⟩.

24. The following screen is displayed:

```
The current result code for 9600 baud connection is:  "CONNECT 9600"
Enter new result code for 9600 baud connection or 'q' to accept this one
-> q
```

    Enter **q** to accept the current result code for 9600 baud connection, or enter a different result code for 9600 baud connection and press ⟨Return⟩.

25. The following screen is displayed:

```
The current result code for 19200 baud connection is:  "CONNECT 19200"
Enter new result code for 19200 baud connection or 'q' to accept this one
-> q
```

Enter **q** to accept the current result code for 19200 baud connection, or enter a different result code for 19200 baud connection and press ⟨Return⟩.

26. The following screen is displayed:

```
The current result code for 38400 baud connection is:  "CONNECT 38400"
Enter new result code for 38400 baud connection or 'q' to accept this one
-> q
```

Enter **q** to accept the current result code for 38400 baud connection, or enter a different result code for 38400 baud connection and press ⟨Return⟩.

27. The following screen is displayed:

```
You now need to complete the dialer installation by moving the
modem specification file into the standard default directory, and by
creating the actual dialer binary.
Do you wish to do this now? (y/n) y
```

Enter **y** to move the modem specification file now, or **n** if you are going to move it later. In either case, control is returned to the command line.

Installation is now complete.

Chapter 16
Backing up filesystems

The main task of a system administrator is to ensure the continued integrity of information stored on the system. Files and filesystems can be damaged and data lost in the following ways:

- power interruptions (make certain you have a surge protector)
- hardware failures (particularly the hard disk)
- user errors (accidental removal of important files)

The importance of having up-to-date backups cannot be overstated. If your system has a number of active accounts, backups require daily attention. It is difficult to estimate the magnitude of a simple loss of data until an accident occurs and several weeks or months of work is gone in an instant.

A filesystem backup is a copy, on storage media (floppy disks or tape) of the files in the root filesystem and other regularly mounted filesystems (for example, the */u* filesystem). (See the "Managing filesystems" chapter in this guide for a discussion of filesystems.) A backup allows the system administrator (or user with the **backup** authorization) to save a copy of a filesystem as it was at a specific time.

This chapter explains how to use **sysadmsh**(ADM) to create backups of the root directory and other filesystems, and how to restore files from the backups. (Another utility used for simple backups, **tar**(C), is discussed extensively in the "Using disks and tapes" chapter of the *User's Guide*. **tar** is not sufficiently sophisticated to perform scheduled backups; it is better suited to archiving groups of files.)

The tools discussed in this chapter present menus with simple options instead of the complex command lines used with the utilities **tar**(C), **cpio**(C), **backup**(ADM), and **restore**(ADM). The key to efficient backups is to save only what has changed from day to day, which (when used with **backup** and **restore**) normally requires extra bookkeeping.

Strategies for backups using sysadmsh

As system administrator, you should familiarize yourself with this chapter and create a backup schedule as instructed. When this schedule is complete, you have only to insert a media volume and respond to a series of prompts to perform your daily backups.

The primary purpose of the **sysadmsh** filesystem backup selection is to provide a dependable schedule of filesystem backups for systems with many users and large filesystems. The program automatically locates modified files and copies them to backup media. If your system has many users and a large number of files that are modified daily, the "scheduled" backup option uses a predefined schedule to make regular backups. When the Backups selection is invoked, the program presents each task as a menu option. To perform a task, simply choose the appropriate option from the menu and supply any required information.

For backups of an informal nature, **sysadmsh** includes an option for "unscheduled" backups. This allows the system administrator to perform a single, complete backup of a filesystem. (Note this type of backup covers the entire filesystem, not just modified files, and may require a number of storage media volumes.) If you intend to rely on unscheduled backups, be sure to perform one at least once a week.

Using the backup authorization

You must assign the **backup** authorization to a user to create or restore backups. (You must be *root* to restore an entire filesystem.) Ordinary users cannot make backups because they do not have access permissions for all files. If backups are made as *root*, files may be accidentally destroyed because *root* has unrestricted permissions on every file on the system. The *backup* authorization solves this dilemma by having restricted root permissions.

Floppy drive backups and large systems

If your system has only a floppy drive, backups for large systems with several users can be time-consuming and use a great deal of media. A complete backup of a 20 Mbyte filesystem requires 15 1.2 Mbyte 96tpi diskettes, while a single 450-foot cartridge tape can store more than twice that amount. More importantly, diskettes require the presence of the operator to insert and remove floppies, whereas a single cartridge tape can be inserted and the operator need not remain by the system. If your system has a large number of users and just a floppy drive, you should install a cartridge tape drive, or make complete system backups once per week and warn your users to make individual backups of their own files on a regular basis.

Summary of utilities accessed

The **sysadmsh** accesses several utilities during the backup process. You do not need to be familiar with them. However, should you wish to use advanced options not discussed in this chapter, you will need to know how they are used and which reference pages to read. **sysadmsh** accesses the following utilities:

- **fsphoto**(ADM) is the main utility that controls the automated backup facilities.
- **fsave**(ADM) is the program that interacts with the user to perform the backup.
- **schedule**(ADM) is the backup database that specifies the media to be used, the filesystems to be backed up, and when to do so.
- **xbackup** and **xrestore**(ADM) are the XENIX backup utilities. These utilities are accessed when "xbackup" appears in the "Method" field of the Schedule table. (These utilities only work for XENIX filesystems.) **cpio** is the preferred method.
- **cpio**(C) is the default backup program. It is non-filesystem specific.

The **sysadmsh** Backups selection forms the "user-friendly" layer that isolates the user from the complicated syntax of these programs.

Preparations for scheduled backups

The only mandatory requirement for scheduled backups is the creation of a backup schedule. In addition, it is recommended that the system administrator follow the optional procedures for labeling, storing, and logging backups. A detailed explanation of backup levels is included at the end of this chapter in case it is necessary to design a more specialized schedule.

Creating a backup schedule

The first step is to create a timetable for backups using the *schedule* file. This file is located in the */usr/lib/sysadmin* directory and contains all the data needed for the system to perform a system backup, including:

- the name of your site or machine
- the media type and drive to be used
- a precise schedule of filesystems to be backed up

The sections that follow explain what changes should be made to the *schedule* file provided with your distribution.

Edit the schedule file

You can edit the *schedule* file with any text editor; make certain you are logged in as *root*. Example 16-1 shows the default *schedule* file. You can also use the following **sysadmsh** selection to edit the *schedule* file:

Backups ➪ Schedule

sysadmsh uses the **vi**(C) editor by default, but you can set the **SA_EDITOR** environment variable to the editor you prefer. See **environ**(M) or **sh**(C) for an explanation of how to set environment variables. The subsections that follow explain the exact changes you need to make to this file.

Example 16-1 The schedule file

```
# SYSTEM BACKUP SCHEDULE
site machinename
# Media Entries
#
# 96 tpi 1.2 MB floppy 0
media /dev/rfd096ds15 k 1200 format /dev/rfd096ds15
# 96 tpi 1.2 MB floppy 1
# media /dev/rfd196ds15 k 1200 format /dev/rfd196ds15
# 135 tpi 1.44 MB floppy 0
# media /dev/rfd0135ds18 k 1440 format /dev/rfd0135ds18
# 135 tpi 1.44 MB floppy 1
# media /dev/rfd1135ds18 k 1440 format /dev/rfd135ds18
# Cartridge tape 1
# media /dev/rct0 k 60000 125000 150000 tape erase
# Mini cartridge drive (10MB)
# media /dev/rctmini k 8800 format /dev/rctmini
# Mini cartridge drive (20MB)
# media /dev/rctmini k 17200 format /dev/rctmini
# Mini cartridge drive (40MB)
# media /dev/rctmini k 37500 format /dev/rctmini
# 9-track tape drive
# media /dev/rmt0 d 1600 2400 1200 600

# Backup Descriptor Table
#   Backup   Vol.   Save for     Vitality        Label
#   level    size   how long     (importance)    marker
    0        -      "1 year"     critical        "a red sticker"
    1        -      "4 months"   necessary       "a yellow sticker"
    2        -      "3 weeks"    useful          "a blue sticker"
    3        -      "1 week"     precautionary   none

# Schedule Table
#               1 2 3 4 5   6 7 8 9 0   1 2 3 4 5   6 7 8 9 0
# Filesystem    M T W T F   M T W T F   M T W T F   M T W T F   Method
  /dev/rroot    0 3 3 3 3   2 3 3 3 3   1 3 3 3 3   2 3 3 3 3   cpio

# Alternate schedule for systems with /u filesystems
#/dev/rroot     0 x 3 x 3   2 x 3 x 3   1 x 3 x 3   2 x 3 x 3   cpio
#/dev/ru        3 0 3 3 3   3 2 3 3 3   3 1 3 3 3   3 2 3 3 3   cpio
```

Add the name of your site or machine

Simply change the ***machinename*** entry at the top of the file to the name you wish.

Select the media device that matches your configuration

Depending on your distribution media, the default drive is either 96tpi 1.2 Mbyte or 135tpi 1.4 Mbyte floppy drive 0, or Cartridge tape 1. The 96tpi drive is reproduced in Example 16-2. The number signs " #" are comment symbols used to "comment out" text so that it is ignored by the program. Note that the default drive is the only one without a comment symbol. If you plan to use a drive other than the default (or a tape drive), put a comment symbol in front of the 96tpi drive and remove the comment symbol from in front of the drive you wish to use. The remaining drives should remain commented out.

NOTE Only one drive can be available at any one time.

Example 16-2 Default media entry

```
# 96 tpi 1.2 MB floppy 0
media /dev/rfd096ds15 k 1200 format /dev/rfd096ds15

# 96 tpi 1.2 MB floppy 1
# media /dev/rfd196ds15 k 1200 format /dev/rfd196ds15
```

Edit the backup descriptor table

Directly below the media drive lines is the Backup Descriptor table. This table, reproduced in Example 16-3, describes each backup level in terms of volume size, how long it is to be stored, how important it is, and how it is marked. The default entries should prove useful, but the volume size entries must be edited according to the type of media you are using.

Example 16-3 Backup descriptor table

```
#   Backup   Vol.   Save for     Vitality         Label
#   level    size   how long     (importance)     marker
    0        -      "1 year"     critical         "a red sticker"
    1        -      "4 months"   necessary        "a yellow sticker"
    2        -      "3 weeks"    useful           "a blue sticker"
    3        -      "1 week"     precautionary    none
```

If you are using floppy disks, leave the dashes in the "Vol. size" column as they are. This causes the backup program to take the volume size from the media entry for that device.

If you are using tapes or tape cartridges, replace each dash in the "Vol. size" column with the size (in kilobytes) of the tape volume. If you are using tapes that are all the same size for each backup level, replace each with the size of the tape you are using.

The last column contains label entries that are discussed in "Labeling your backups" later in this section.

Edit the backup schedule table

The default schedule assumes that backups are done every day. To make backups more efficient, they are broken into levels. Level 0 is the lowest level backup. A level 0 backup saves everything on the filesystem, while 1, 2, and 3 each back up only the files that have changed relative to the last lower-level backup. This concept is illustrated in Figure 16-1 with a stack of toy disks representing each level. Notice that the level 0 disk is the largest backup, and each of the others is progressively smaller. This is because level 1 contains only the files that were changed since the level 0 was done, and so on. This figure also illustrates how these backups would be restored: first the level 0, followed by each of the latest 1, 2, and 3 that were done.

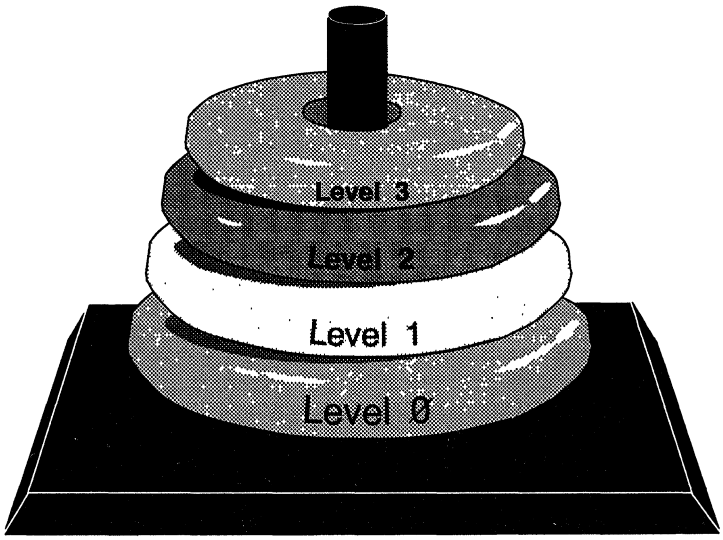


Figure 16-1 Backup levels

The concept of levels may seem needlessly complex at first, but consider what would happen on a system with a number of large filesystems. If you performed a full backup of each filesystem each night, the process would take hours to perform, bogging the system down in the process. If only the files that changed most recently are saved, backups would be less time-consuming and, depending on the size of your media, consume fewer volumes. (See "An explanation of backup levels" later in this chapter for a more detailed discussion of backup levels.)

The example *schedule* file in this chapter includes an alternate schedule for a */u* filesystem that is commented out. Note that there is a backup done every other day for the root filesystem and once a day for the */u* filesystem. This is because the */u* filesystem (user accounts) changes much more frequently than the root filesystem, which contains the system files. An "x" means that a backup is not performed on that day for that filesystem.

If you do not have a */u* filesystem, then your user accounts are located in the root filesystem (in the directory */usr*). If this is so, the schedule table is preconfigured to back up the root filesystem. However, if you have added a */u* filesystem, edit the schedule table and remove the # in front of the entry for */dev/ru,* shown in Example 16-4. This ensures that backups are made of the additional filesystem. If you do not have a */u* filesystem, but you do want daily backups, this entry can also be modified and used for the root filesystem.

Example 16-4 Backup schedule table

```
#              1 2 3 4 5   6 7 8 9 0   1 2 3 4 5   6 7 8 9 0   Method
# Filesystem   M T W T F   M T W T F   M T W T F   M T W T F
  /dev/rroot   0 x 3 x 3   2 x 3 x 3   1 x 3 x 3   2 x 3 x 3   cpio
  /dev/ru      3 0 3 3 3   3 2 3 3 3   3 1 3 3 3   3 2 3 3 3   cpio
```

Note that the Monday-Friday notation can be misleading; if a backup is postponed or unsuccessful (because of bad media, for example) then that same level backup is attempted again at the next scheduled backup. This offsets the schedule, but does not alter the established sequence of backups. The numbered scale of 1-0 above M-F is more accurate, but less useful to people, who work in day and week units.

In addition, if you add lines for other filesystems, you should take care not to schedule two level 0 backups of large filesystems on the same day; the process is lengthy and may slow your machine significantly.

Backup method field

The default backup format type is **cpio**. If you wish to use the **xbackup** format used with XENIX filesystems, replace the "cpio" with "xbackup". Note that the **xbackup**(ADM) utility only works on XENIX filesystems because it uses filesystem-specific information. The **cpio** format functions perfectly with UNIX, XENIX, AFS, and EAFS filesystems because it does not use such information. Use of the **xbackup** type is not recommended.

NOTE Remember that EAFS is the default filesystem type used by the operating system.

Labeling your backups

It is important to label your backup tapes with meaningful and accurate information. If your backups consist of a pile of haphazardly labeled tapes, it will be difficult to locate data at a later date.

Figure 16-2 is a suggested format for media labels.

Name of computer	Backup level Filesystem Name save until date	Date made
Name of backup person		volume # of #

Figure 16-2 Sample media label

The date on the label, and the date from which you calculate the "save until" date, should be the date of the business day covered by the backup. This is to avoid confusion if it becomes necessary to restore information from this tape.

You may have noticed that the *schedule* file has a proposed color-coding scheme for easy reference, as emphasized in Example 16-5.

Example 16-5 Backup labeling scheme

```
#   Backup   Vol.   Save for     Vitality        Label
#   level    size   how long     (importance)    marker
    0        -      "1 year"     critical        "a red sticker"
    1        -      "4 months"   necessary       "a yellow sticker"
    2        -      "3 weeks"    useful          "a blue sticker"
    3        -      "1 week"     precautionary   none
```

If there is more than one tape for a single backup, mark the date label on each volume to indicate the volume number and number of volumes, such as "1 of 2" and "2 of 2" for a two-volume backup. Finally, place a label on the side of the box or enclosure marked with the name of the computer, the filesystem, and the backup level completed.

Keeping a log book

It is recommended that a written log book be maintained for each computer. In addition to maintenance information (such as when breakdowns occur and what was done about it), you should record the following information:

Date	Just as with the tape label, this date should be the last day covered by the backup.
Filesystem	This is the name of the device backed up on the current tape.
Backup level	This is the backup level of the current tape.
#Vols	This is the number of tape volumes.
Start/finish time	(Optional.) The time from the start of a backup of a filesystem until the last error check is completed. The times are displayed after the backup is finished. The finish time will often be inaccurate, since you may be out of the room when the backup finishes, and the machine sits idle before you return.

If there are problems with the backup, record these in the log book as well, including any error messages that come to the screen.

Rotating backup media

Backup media should be used so that at least 6 to 12 months of media are left on file. The default *schedule* file includes a suggested rotation of 1 week on level 3 backups; 3 weeks on level 2; 4 months on level 1; and 1 year on level 0. This means that if you follow the default schedule, you can safely re-use your level 3 backups after 1 week, and so on.

Archiving backup media

All filesystems should be periodically backed up and archived offsite. In the event of a fire or natural catastrophe, the data can later be restored.

Performing a scheduled backup

This section describes how to perform a backup using a defined schedule. Do not attempt this until you have edited (or at least examined) the *schedule* file to make certain that it suits your needs.

The system administrator should schedule backups at times when few (if any) users are on the system. This ensures that the most recent version of each file is copied correctly.

A regular schedule of backups requires a good supply of media and adequate storage for them. Level 0 backups should be saved at least a year, longer if they are important. Lesser backups should be saved at least two weeks. Media volumes should be properly labeled with the date of the backup and the names of the files and directories contained in the backup. After a backup has expired, the media can be used to create new backups.

Using formatted media

If you use media that requires formatting, such as floppy disks or tape cartridges, you are advised to always have formatted volumes before you begin. The exact number of volumes depends on the number and size of files to be backed up. For details on how to format your media, see the "Using floppy disks and tape drives" chapter in this guide. You also have the option to do formatting from the **sysadmsh** program, but you cannot format media while a backup is in progress.

NOTE Formatting tape cartridges is not recommended. Best results are achieved with preformatted cartridges.

Starting the backup

To run your scheduled backup, follow these steps:

1. Invoke **sysadmsh** and select the following:

   Backups ⇨ Create ⇨ Scheduled

2. A menu is displayed that looks like the following:

```
Level 0 backup of filesystem /dev/rroot, 31 Aug
          tape size:          1200 Kb
          tape drive:         /dev/rfd096ds15
This tape will be saved for 1 year, and is critical.

M)ounted volume, P)ostpone, C)heck or F)ormat volumes, R) Retension or H)elp:
```

The media type displayed is the one entered in the *schedule* file. Load a volume, tape or disk, into the selected drive. Enter **m** to tell the program the volume is mounted, and press ⟨Return⟩.

3. The system displays the current date and the date of the last backup:

```
Level 0 backup of filesystem: /dev/rroot
Backing up all files
Generating list of pathnames for backing up ...
```

This process takes a few minutes.

4. The system then begins to copy files to the drive. If a volume runs out of space, the program displays the following messages:

```
Reached end of medium on output
Insert volume 2 and press ⟨Return⟩ to continue or 'q' to exit.
```

NOTE If **xbackup**(ADM) is being used to make the backup, a slightly different prompt is displayed.

Remove the present volume, insert a new volume, then press ⟨Return⟩. The program continues to copy files to the new volume. Repeat this step until the program displays the message:

```
Check critical volumes for format errors
```

5. When the backup is complete, the following menu is displayed:

```
M)ounted which volume, S)kip format check, or H)elp
```

Level 0 backups should always be checked for format errors. Enter **m** to check your media. If you are checking the format, make certain you insert the first volume as instructed, or the backup aborts. If you do not want to check the volumes, enter **s**.

6. If an error occurs, the backup is declared unsuccessful and is retried from the beginning. Your media could be bad, so replace it if errors persist. The menu keeps track of the volume being checked:

```
M)ounted which volume, E)rror on previous volume, D)one,
S)kip checks, or H)elp:
```

When you are finished checking volumes, select **d**.

After the backup is successfully performed, instructions are given on how to label the volumes. Make certain that you write-protect your volumes.

Performing an unscheduled backup

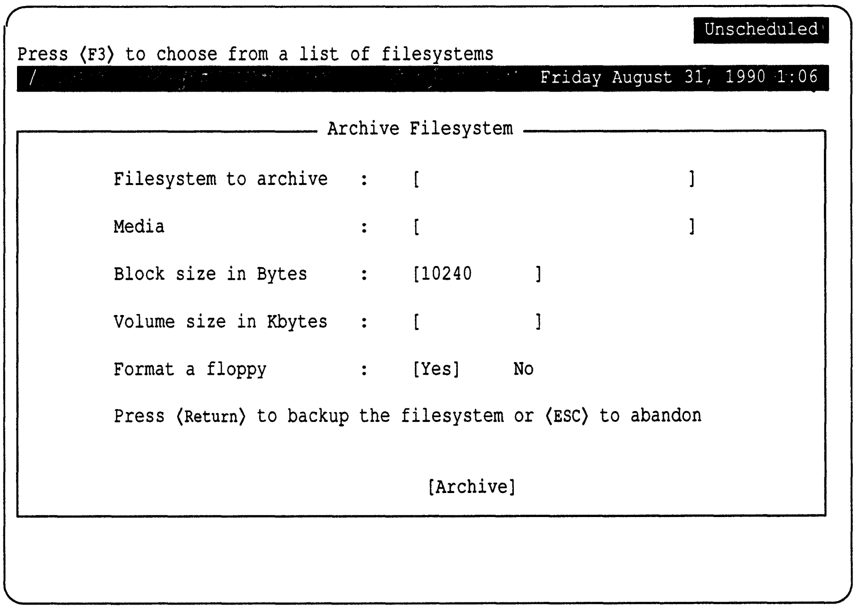
For backups of an informal nature, **sysadmsh** includes an option for "unscheduled" backups. This allows the system administrator to perform a single, complete backup of a filesystem without using a schedule. If your backup needs are simple, you can do an unscheduled backup on a regular basis. This type of backup covers the entire filesystem, not just modified files, and may require a number of storage media volumes. If you intend to rely on unscheduled backups, be sure to perform one at least once a week.

To create an unscheduled backup, follow these steps:

1. Invoke **sysadmsh** and select the following:

   Backups ➪ Create ➪ Unscheduled

2. The following menu is displayed:



```
                                                              Unscheduled
Press (F3) to choose from a list of filesystems
/                                            Friday August 31, 1990 1:06

─────────────────────────── Archive Filesystem ───────────────────────────

        Filesystem to archive   :   [                          ]

        Media                   :   [                          ]

        Block size in Bytes     :   [10240       ]

        Volume size in Kbytes   :   [            ]

        Format a floppy         :   [Yes]    No

        Press (Return) to backup the filesystem or (ESC) to abandon


                                  [Archive]
```

3. Select the filesystem to back up by entering the name or pressing ⟨F3⟩ to get a point-and-pick list. The menu lists all filesystems found in the file */etc/default/filesys* (see **filesys**(F)). Use the arrow keys to select the filesystem you wish to back up and press ⟨Return⟩.

4. Next, select the media device to be used by entering the name or pressing ⟨F3⟩ to get a list. The block size is selected automatically.

   NOTE Take care when selecting the number of the media device. For example, make certain that you do not select "Floppy Drive 1" (the secondary floppy drive) when you want "Floppy Drive 0" (the primary floppy drive). If you make this error, the backup is aborted and you must start over.

5. You can format as many volumes as you wish by inserting each volume one at a time into the drive and selecting Yes on the Format floppy. (Cartridge tapes can also be formatted, but this takes a great deal of time.)
6. Load a volume, tape or disk, into the selected drive, and press ⟨Return⟩. The system then begins to copy files to the drive, displaying the filenames as they are backed-up. If a volume runs out of space, the following is displayed :

```
Reached end of medium on output
Insert volume 2 and press ⟨Return⟩ to continue or 'q' to exit.
```

7. Remove the first volume, insert a new volume, then press ⟨Return⟩. The program continues to copy files to the new volume. Repeat this step until the program displays the message:

```
DONE
```

If you are using floppies, you may need to repeat the last step several times before the backup is complete. You should label each volume as you remove it from the drive. For example, label the first volume "Volume 1", the second "Volume 2", and so on.

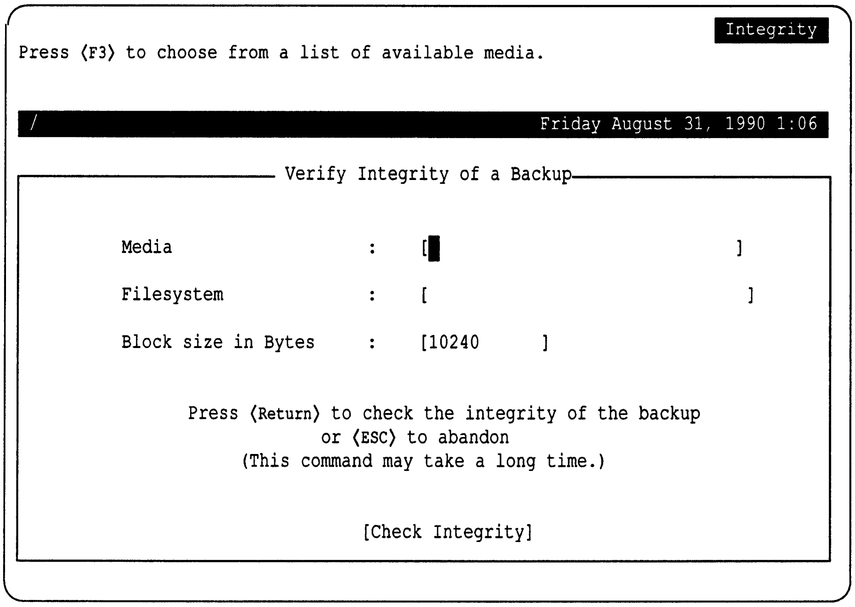
Verifying a backup

To ensure that your backup volumes are accurate and error-free, the **sysadmsh** backup menu includes an Integrity option. The volumes are checked to see if they are readable and the contents are listed.

Invoke **sysadmsh** and select the following:

Backups ⇨ Integrity

The following form is displayed:



```
                                                          Integrity
Press <F3> to choose from a list of available media.

 /                                         Friday August 31, 1990 1:06

 ───────────────────── Verify Integrity of a Backup ──────────────────

        Media                   :    [                        ]

        Filesystem              :    [                         ]

        Block size in Bytes     :    [10240      ]

              Press <Return> to check the integrity of the backup
                              or <ESC> to abandon
                        (This command may take a long time.)

                               [Check Integrity]
```

Enter the Media type, or press ⟨F3⟩ to select it from a point-and-pick list. When selected, a window pops up to confirm the drive is ready:

```
                    Checking type of backup...
               Make sure the media is in the drive
                    and the drive is on line.

         Press <Return> to continue or <ESC> to abandon
```

Insert each volume of the backup in turn. This is a lengthy process which may take some time.

Getting a backup listing

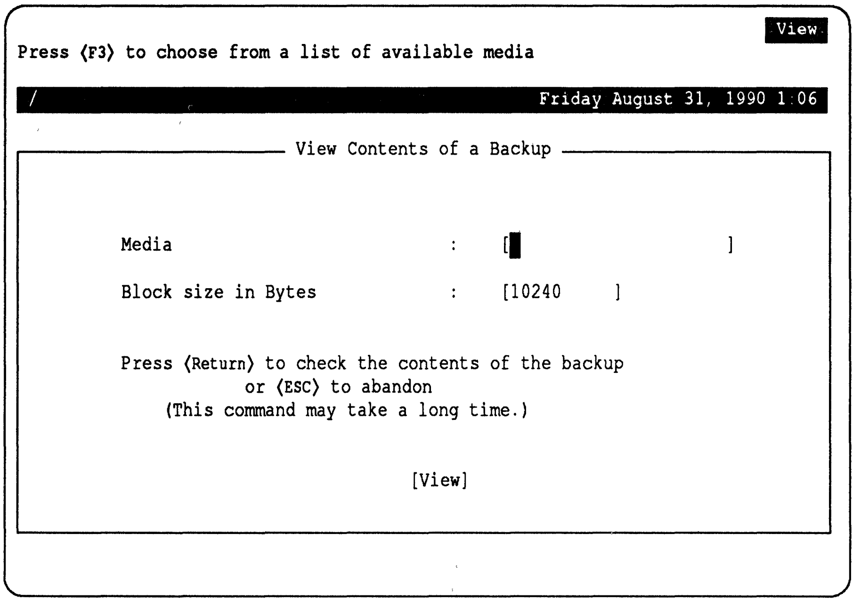
You can examine a list of the files you have backed up by generating a listing from the **sysadmsh** Backups menu.

To get the listing, follow these steps:

1. Select the following from **sysadmsh**:

   Backups ➪ View

2. The following form is displayed:



```
                                                                    View
Press <F3> to choose from a list of available media

 /                                                Friday August 31, 1990 1:06

 ─────────────────────── View Contents of a Backup ─────────────────────────


          Media                          :    [                    ]

          Block size in Bytes            :    [10240      ]


          Press <Return> to check the contents of the backup
                      or <ESC> to abandon
                 (This command may take a long time.)


                                     [View]
```

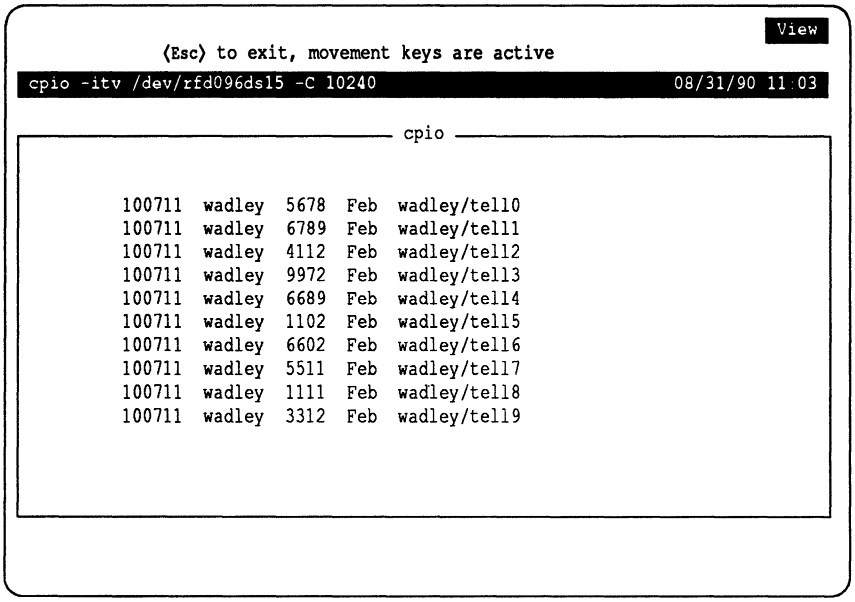
3. Press ⟨F3⟩ at the first field to get a listing of media devices. When you select a media device, a window pops up to confirm the drive is ready:

```
                     Checking type of backup...
               Make sure the media is in the drive
                     and the drive is on line.

          Press <Return> to continue or <ESC> to abandon
```

The block size is selected automatically.

4. The program prompts you to insert each backup volume in turn.

5. When all volumes of the backup are read, a screen similar to the following is displayed:



```
                    <Esc> to exit, movement keys are active
cpio -itv /dev/rfd096ds15 -C 10240                          08/31/90 11:03

                                   cpio

        100711  wadley  5678  Feb  wadley/tell0
        100711  wadley  6789  Feb  wadley/tell1
        100711  wadley  4112  Feb  wadley/tell2
        100711  wadley  9972  Feb  wadley/tell3
        100711  wadley  6689  Feb  wadley/tell4
        100711  wadley  1102  Feb  wadley/tell5
        100711  wadley  6602  Feb  wadley/tell6
        100711  wadley  5511  Feb  wadley/tell7
        100711  wadley  1111  Feb  wadley/tell8
        100711  wadley  3312  Feb  wadley/tell9
```

Restoring individual files or directories from backups

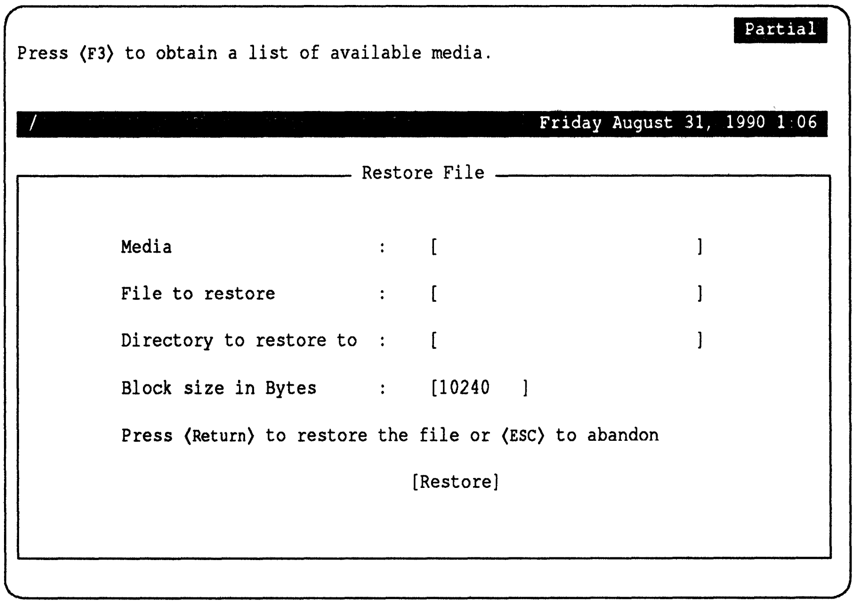
You can restore individual files or directories from your filesystem backup volumes by invoking **sysadmsh**. You need the complete set of backup volumes containing the latest version of the file or files you wish to restore. If you are restoring a file that was not changed recently, use the last level 0 backup.

To restore a file, follow these steps:

1. Invoke **sysadmsh** and select the following:

   Backups ⇨ Restore ⇨ Partial

2. You see the following:



```
                                                              Partial
Press <F3> to obtain a list of available media.


 /                                         Friday August 31, 1990 1:06

                       Restore File

      Media                    :   [                      ]

      File to restore          :   [                      ]

      Directory to restore to  :   [                      ]

      Block size in Bytes      :   [10240   ]

      Press <Return> to restore the file or <ESC> to abandon

                                  [Restore]
```

3. Press ⟨F3⟩ first to select the Media type from a point-and-pick list. When selected, a window pops up to confirm the drive is ready:

```
                 Checking type of backup...
             Make sure the media is in the drive
                 and the drive is on line.

         Press <Return> to continue or <ESC> to abandon
```

4. Load volume 1 of the backup set into the drive, then press ⟨Return⟩. When this request is satisfied, you are returned to the "Restore File" menu. Enter the filename next, then press ⟨Return⟩ to move to the "Directory" field, entering the directory you wish to restore the file(s) to.

   NOTE Two important points:

   - When specifying the pathname, the leading slash "/" must be removed. For example, if you are restoring the file */bin/foo*, you must specify it like this:

     bin/foo

   - If you respond with the pathname of the original location, the restored files overwrite any files by the same names in that location. It is important to be sure that the files on the backup volume are the desired versions of these files. If you are not absolutely sure that your backup contains the preferred version of the files, you should restore them to a temporary location, such as */tmp*, and compare them with your current files on disk using **diff**(C) or **cmp**(C).

5. The archive is searched for the files specified and the filename is displayed after it is restored to the specified locations on your hard disk. You are also prompted to switch volumes if necessary. If you know all the files you want were restored, you can exit the restore using the ⟨Del⟩ key. (The program continues to search to the end of the backup.)

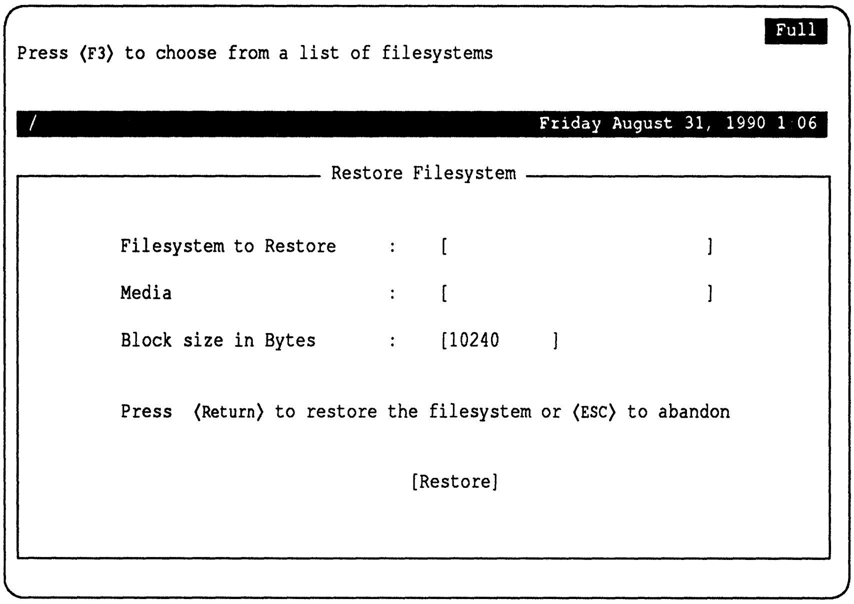
Restoring an entire filesystem

Follow these steps to restore your filesystem backup:

1. Insert the first volume, and make the following **sysadmsh** selection:

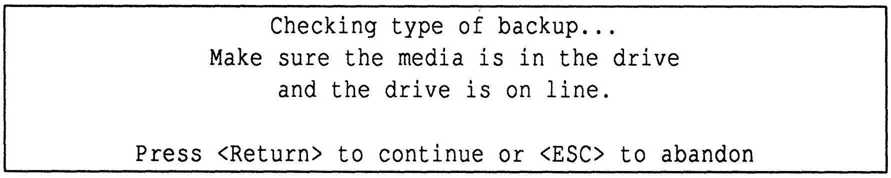
   Backups ⇨ Restore ⇨ Full

   The following form is displayed:



```
                                                                      Full
Press ⟨F3⟩ to choose from a list of filesystems


 /                                            Friday August 31, 1990 1 06

 ─────────────────────────── Restore Filesystem ──────────────────────────

         Filesystem to Restore     :   [                    ]

         Media                     :   [                    ]

         Block size in Bytes       :   [10240     ]

         Press  ⟨Return⟩ to restore the filesystem or ⟨ESC⟩ to abandon

                                 [Restore]
```

2. Enter the name of the filesystem, or press ⟨F3⟩ for a point-and-pick list. Do the same for the media device. The following window pops up to confirm the drive is ready:



```
                 Checking type of backup...
             Make sure the media is in the drive
                  and the drive is on line.

         Press <Return> to continue or <ESC> to abandon
```

3. You are asked to confirm that this is what you wish to do.

4. As each file is restored, the name is printed on the screen. If your backup has multiple volumes, you are prompted to insert each in turn:

```
Reached end of medium on input
Change to part n and press <Return> key. [q]
```

NOTE If **xrestore** is used to restore the filesystem, a slightly different prompt is displayed.

When the restoration process is complete, the number of blocks restored is displayed.

An explanation of backup levels

The most straightforward and dependable way to ensure the safety of data is to back up everything on a filesystem at one time. However, filesystems can be large (as much as 200 Mbytes or more), and may take hours to back up. The concept of backup levels (or incremental backups) addresses this problem. The general idea of an incremental backup is to back up only those files that have changed since a previous backup. This can significantly reduce the size and duration of the backup. Consider the following scheme:

Monthly	complete backup
Weekly	everything newer than last week
Daily	everything newer than yesterday

This means that at the end of every month, the entire filesystem is backed up. Each week, the files that have changed since last week are backed up, and each day, any files that have changed since yesterday. If at some point a filesystem is damaged, you would simply restore the last full (monthly) backup, the last weekly backup, and any daily backups that happened just prior to the accident. Thus it is always possible to reconstruct a filesystem from a series of backups.

While this is a simple method to understand, the implementation using incremental backup levels is not.

Principles of incremental backup levels

To make the business of backing up files more efficient, the backup facility uses a progressive series of levels, each of which is based on the last occurrence of a lower-level backup.

Level	Files saved
0	all files on the filesystem
1	files changed since last level 0 backup
2	files changed since last level 1 backup
3	files changed since last level 2 backup

The levels serve to subdivide a backup into manageable units. It is important to realize that each backup level creates backups based on the previous (next lowest) level backup. This means that the order of the backups is not significant, but the level number is.

For example, let's assume that the following backups were done for a week:

Day	Level	Files backed up
Mon	0	all files on filesystem
Tue	2	all files changed since Monday
Wed	1	all files changed since Monday
Thu	3	all files changed since Tuesday
Fri	2	all files changed since Wednesday

This example is illogical, but serves to demonstrate how the levels work. Remember that each of the backups saves the files changed since the next lower-level backup, and that level 0 is the lowest. Therefore, the level 2 on Friday backs up all files changed since the next lowest number, level 1, on Wednesday. The level 2 on Tuesday saves only those files that have changed since the day before, since the only previous lower-level backup is a 0. If all the backup levels except Monday were level 2, each would still back up all files that changed since the level 0 on Monday.

How the default and alternate schedules work

The *schedule* file provided with your distribution is optimized for use on systems under moderate use (8 to 10 users with total disk storage of 200 to 400 Mbytes). The default schedule for the root filesystem is similar to that used for the */u* filesystem in the alternate schedule. This is done because a system with a single filesystem (*root*) has active user accounts and should be backed up each day. A system with a second filesystem (*/u*) for user accounts is backed up each day, while the less active root filesystem is backed up every other day.

The alternate schedule is shown in Example 16-6.

Example 16-6 The default schedule

```
#               1 2 3 4 5   6 7 8 9 10   1 2 3 4 5   6 7 8 9 10
#   Filesystem  M T W T F   M T W T F    M T W T F   M T W T F
    /dev/rroot  0 x 3 x 3   2 x 3 x 3    1 x 3 x 3   2 x 3 x 3
    /dev/ru     3 0 3 3 3   3 2 3 3 3    3 1 3 3 3   3 2 3 3 3
```

The /u filesystem

Filesystem */dev/u* is a heavily used resource. Some level of backup is performed every day. This scheme is designed to minimize resources while maximizing safety; if one or more of the backups for that week is lost or goes bad, there is sufficient redundancy to minimize any loss of data.

According to the default schedule, a full (level 0) backup of */dev/ru* occurs at the beginning of the month. (Because a level 0 is done on the root filesystem on Monday, the level 0 for */u* is done on Tuesday.) On Wednesday, a level 3 backup saves just those files on */dev/ru* that have changed since the level 0 backup. By the end of the week far fewer floppies or tapes are used than the number needed for full backups each day. Time is substantially reduced as well. If it is necessary to restore the filesystem to the last recorded state, you would restore the last level 0 backup, followed by each of the most recent lower-level backups that were done since.

Note that each Tuesday, a lower-level backup (0, 1 or 2) occurs that saves everything since the beginning of the month and causes each of the level 3 backups that follow it to be based on that week. This way the level 3 backups do not become too large and redundant.

The root filesystem

The root filesystem contains the operating system and other system files. It changes less frequently, so it is not backed up every day unless user accounts are located there. Each Monday, a lower-level backup is done, and level 3 backups are done twice per week. Just as with the */u* filesystem, the level 3 backups are restricted to cover only those files that have changed during that week.

How backups restore a filesystem

For example, assume you have a hardware failure that ruins the information on the hard disk. Assume it happens on the last Thursday of the month, just before the backup was to be done that evening. You fix the hardware problem and reinstall your system, but how do you restore your backups? Restore the last occurrence of each backup level, in ascending order:

- level 0 (done on the first Tuesday of the month)
- level 1 (done on the third Tuesday)
- level 2 (done on the fourth Tuesday)
- level 3 (done on Wednesday evening)

You would not need to restore the level 2 that was done on the second Tuesday, because the level 1 that followed it covered the same files. The only information that is missing is what was changed during the day on Thursday, just before the crash. This is the primary reason for backups; recovery should be straightforward and with a minimum of loss.

Unattended backups

The methods for performing backups described so far require human operation. You may find it more convenient for your backups to be performed when the system is unattended, during early mornings for example. This can be achieved by making entries in root's *crontab* file to call the **cbackup** shell script at the desired times. If you are unfamiliar with the **cron** daemon and its use then examine the **cron**(C), and **crontab**(C) manual pages for an explanation of its function.

The cbackup shell script

This shell script is the last component that **fsphoto**(ADM) calls to perform a **cpio** backup. Using **cbackup** directly will bypass the schedule file, but still allow you to make incremental backups. **cbackup** takes four arguments:

level:	The increment level of the backup you wish to make.
size:	The capacity of one volume of the media you are going to record the backup on.
device:	The name of the device to record the backup on. You should always specify a raw device.
filesystem:	The name of the device of the filesystem to be backed up. **cbackup** looks up the mount point of the filesystem in */etc/default/filesys*. Since **cbackup** uses **cpio**, the filesystem must be mounted when it is backed up.

NOTE A single day's backups must fit on one volume for unattended backups to be successful.

Example crontab entries

If, for example, you wish to backup the entire root filesystem at 2.00am every day on 150MB cartridge tape, the following root *crontab* file could be used:

```
* 2 * * * /usr/lib/sysadmin/cbackup 0 150000 /dev/rct0 /dev/root
```

If you wish to perform a level 1 backup of the root filesystem every weekday morning and a full backup on Saturday morning then the following root *crontab* file could be used:

```
* 2 1,5 * * * /usr/lib/sysadmin/cbackup 1 150000 /dev/rct0 /dev/root
* 2 6   * * * /usr/lib/sysadmin/cbackup 0 150000 /dev/rct0 /dev/root
```

NOTE As mentioned before, **cbackup** ignores the schedule file. A hierarchy of incremental backups can be constructed by stating explicitly in the *cron tab* file when a particular level should be performed.

If you wish to back up more than one filesystem to the same unattended device, then the backups have to run sequentially. To ensure this happens it is best to place the calls to **cbackup** in a shell script and make a **crontab** entry to call it. For example, if you wish to perform a level 1 backup of the */u* and *root* filesystems to a cartridge tape every weekday night then the following could be used:

crontab entry:

```
* 2 1,5 * * * /usr/lib/sysadmin/bscript
```

/usr/lib/sysadmin/bscript:

```
/usr/lib/sysadmin/cbackup 1 150000 /dev/nrct0 /dev/root
/usr/lib/sysadmin/cbackup 1 150000 /dev/nrct0 /dev/u
tape rewind /dev/xct0
```

Notice the use of the no rewind tape device (*/dev/nrct0* here). It is required so that when the backup of the *root* filesystem has finished the tape is not automatically rewound to the beginning. For convenience you may wish to rewind the tape after the backup of */u* has finished.

Chapter 17
Tuning system performance

Your UNIX system is optimized for use with a variety of hardware configurations and as a platform for many applications. The kernel, which lies at the heart of the operating system, controls a number of resources that are constantly being used, released, and recycled. These resources include:

buffers	A cache of in-memory storage units that hold recently used data. (Buffers increase efficiency by keeping this data on hand and decrease reading from the disk.)
table entries	A space in system tables that the kernel uses to keep track of current tasks, resources, and events.
other parameters	These are other definable values that govern special resources (such as the number of multiscreens available or the quantity of semaphores).

The use of these resources is defined by certain limits known as *tunable kernel parameters*. These limits can be decreased or extended, sometimes at the expense of other resources. Each resource or limit is represented by a separate kernel parameter. Deciding how to best optimize the use of these resources is known as performance or kernel tuning. This chapter explains how to change these parameters to suit the needs of your system. In addition, general procedures are included that can improve resource usage and system performance.

Kernel parameters

Kernel parameters are values contained in the UNIX system kernel, which is the core of the operating system. Each time these tunable parameters are changed, their new values are relinked (recompiled) into the kernel so that the new limits will take effect. The **configure**(ADM) utility changes the value of kernel parameters.

Performance tuning is an activity that may need your attention when you first set up your UNIX system. When you bring the system up for the first time, the system is automatically set to a basic configuration that is satisfactory for most sites. This configuration, however, cannot take into account the usage patterns and the behavior of your particular applications. For this reason, the structure of the system allows you to reconfigure it to enhance the performance for your particular application over that of the standard configuration.

NOTE We do not recommend adjusting kernel parameters if there is no apparent need to do so.

There are several reasons for reallocating system resources:

- You install additional hardware memory and thus have greater memory resources to allocate.
- Persistent error messages are displayed indicating that certain resources are used up, such as inodes or table entries.
- The system response time is consistently slow, indicating that other resources are too constrained for the system to operate efficiently (as when too little hardware memory is installed).
- Resource usage needs to be tailored to meet the needs of a particular application.

In addition, it is important to determine which resources are being wasted or are inefficiently distributed. Certain tunable parameters are normally adjusted upward when additional memory is installed to allow the system to support more users. However, for a computer used as a high-powered personal computer, or dedicated processor, it may not be necessary to increase kernel tunable parameters when additional memory is installed. In fact, tuning certain parameters normally associated with adding memory to support more users (**NBUF**, **NCLIST**, and so on) can actually decrease overall performance. This is because these parameters increase kernel data space requirements, thus making less of the new memory available for user processes. Simply stated, the intended use of your computer and your observations on how well it is performing should be used as a guide in determining the need to adjust tunable parameters.

Specialized applications often require the reallocation of key system resources for optimum performance. For example, users with large databases may find that they need to lock more files simultaneously than the current allocation of file locks permit. Users who have no need for specialized features such as message handling may find that they can get a slight performance boost by deallocating those features.

Reallocating kernel resources with configure

The **configure** utility is a menu-driven program that presents each resource and prompts for modification. After modifying kernel parameters, you must relink the kernel by invoking **link_unix**(ADM), copy the kernel to the root directory, reboot, and test the new kernel.

NOTE This section only describes how a parameter change is made; you must read the rest of the chapter to understand why and when a change must be made.

To change any kernel parameter, do the following:

1. Reboot and enter single-user (maintenance) mode.
2. Use **custom**(ADM) to determine if the link kit package (LINK) is installed. If not, use **custom** to install it.
3. After making certain the Link Kit is installed, enter the following commands:

   ```
   cd /etc/conf/cf.d
   ./configure
   ```

   Δ **sysadmsh** users select: System ➪ Configure ➪ Kernel ➪ Parameters
4. The **configure** menu is displayed:

```
1.       Disks and Buffers
2.       Character Buffers
3.       Files, Inodes, and Filesystems
4.       Processes, Memory Management and Swapping
5.       Clock
6.       MultiScreens
7.       Message Queues
8.       Semaphores
9.       Shared Data
10.      System Name
11.      Streams Data
12.      Event Queues and Devices
13.      Hardware Dependent Parameters
14.      Security
15.      Asynchronous I/O

Select a parameter category to reconfigure by
typing a number from 1 to 15, or q to quit:
```

The parameters are grouped by category. To locate a parameter, see "Tunable system parameter descriptions" at the end of this chapter. Choose a category by entering the number preceding it. The resources in that category are displayed, one by one, each with its current value. Enter a new value for the resource, or to retain the current value, simply press ⟨Return⟩. After all the resources in the category are displayed, **configure** returns to the category menu prompt. Choose another category to reconfigure or exit **configure** by entering **q**.

NOTE Note that you must have the package associated with the parameter to make use of the resource associated with it. For example, the STREAMS package must be installed for the STREAMS parameters to have effect.

5. After you finish changing parameters, you must link them into a new kernel. Enter the following command:

   ```
   ./link_unix
   ```

   Δ **sysadmsh** users select: System ➪ Configure ➪ Kernel ➪ Rebuild

   This assembles each of the kernel modules into a new kernel, which must now be installed. Linking can take a while.
6. Boot the new kernel with the following command:

   ```
   /etc/shutdown
   ```

   Δ **sysadmsh** users select: System ➪ Terminate

   A boot prompt appears. When you press ⟨Return⟩ to reboot the system, the new kernel is loaded and run.

If problems exist with the new kernel, reboot */unix.old* by entering **unix.old** at the Boot: prompt.

Using the configure command line

configure also has a command-line interface suitable for use by application developers. For instance, a database developer who finds that 70 files rather than 50 files need to be locked simultaneously may provide a shell script to perform the reconfiguration. To find the current value of any configurable resource using the command-line interface, enter:

```
./configure -y RESOURCE
```

where *RESOURCE* is the name of the tunable parameter (in uppercase).
To change the value of any resource from the command line, enter:

```
./configure RESOURCE=value
```

The sections that follow describe scenarios for reconfiguring the kernel resources.

Overriding configure limit warnings

The **configure** utility enforces certain limits on values for kernel parameters. This is because when certain values are exceeded, system performance can suffer dramatically if it is done by mistake. If you find that you must override a parameter value, use the **-o** option as described on the **configure**(ADM) manual page. The override option only works if you are specifying a parameter on the command line; you cannot use it with the menu.

Reconfiguring because of persistent error messages

There are cases when the operating system advises you that system limits are being exceeded. These messages are displayed on the console. Some of the messages are advisory only. Others precede a system panic in which case additional diagnostic messages are printed, and the system "hangs," requiring you to reboot. The kernel should not be reconfigured because a kernel error message was received once, or even a couple of times, but when a single message persists between system sessions.

If you encounter any of these messages refer to the descriptions for the appropriate parameters in the section "Tunable system parameter descriptions" later in this chapter. If you need to adjust a resource, first try to increase the value by a small amount. If the problem persists, increase it by 100 percent or more of its original value. If the problem is still not solved, more detailed research is required to locate the exact program and sequence that cause the error. The format of the error messages is:

CONFIG: ***routine* - *message*** (***parameter* = *value*** exceeded)

For example:

```
CONFIG: timeout - Timeout table overflow (NCALL = n exceeded)
```

where ***n*** is the actual value displayed in the error message.

Reconfiguring for performance

The system is configured so that the greatest quantities of kernel resources are assigned to the most common tasks (such as reading and writing from the disk), without ignoring the more specialized features (such as interprocess communication). This balance can be shifted to conform to individual requirements using the information described in the sections that follow.

Tradeoffs in kernel tuning

Systems can support very different usage of resources. A system that supports several users editing small files consumes different resources than a single-user system running a large database. Parameters are adjusted to allow the kernel to operate more efficiently. This often increases the size of kernel data structures.

Although this might make one aspect of system operation more efficient, the kernel takes longer to scan larger structures. This implies that increasing certain parameters unnecessarily can actually slow the system down. For example, increasing the parameter **NPROC** allows the system to maintain a larger list (PROCess table) of active processes. This can have an adverse effect on the kernel scheduler because it must now repeatedly scan this larger table every time it checks to see which process to run next. Additionally, because the kernel data space requirements increase when table sizes are increased, there is less memory space available for user processes, which can also lower overall performance.

Common resource needs

Often your system usage presents you with the need to tune certain parameters for particular circumstances. A common need is the ability to create very large files. This can be accomplished by becoming the super user and modifying the ulimit for the particular shell process that you are running as super user. An alternate solution is to modify the system **ULIMIT** for all users. The **ULIMIT** parameter and other commonly encountered limits are summarized in Table 17.1. Refer to the section "Tunable system parameter descriptions" in this chapter for details on each parameter before you make any alterations.

Table 17-1 Special case tuning needs

Desired improvement	Parameters
Improve system performance when additional memory is installed.	NBUF, NHBUF (see "Sizing the buffer cache")
Increase system limits when additional memory is installed (to support more users and reduce chances of system problems at times of heavy load, and so forth).	NCALL, NINODE, NSINODE, NFILE, NPROC, NREGION, NCLIST (also see message, semaphore, and shared memory parameters)
Users need to create bigger files.	ULIMIT
Each user needs to open more files.	NOFILES
Each user needs to run more processes.	MAXUP
Other system limits that may be encountered	SHLBMAX, FLCKREC, SPTMAP, NUMXT, NUMSXT, PRFMAX (also see STREAMS parameters)

Improving disk utilization

Disk input and output can cause a bottleneck in system performance. There are three considerations in tuning the disk subsystem for better utilization:

- choosing the proper number of buffers
- adding more memory
- organizing the filesystems to minimize disk activity

Sizing the buffer cache

The system effectively divides available memory between two structures: the disk buffer cache and the page cache. The buffer cache is a series of buffers that hold recently used data in case it is needed again. If a read or a write can be satisfied using the buffer cache instead of the disk, system performance improves because memory operations are much faster than disk operations. The page cache is similar in concept to the buffer cache; disk buffers contain data while pages contain programs.

Ideally, you should install sufficient memory for the amount of work done on the system, but if memory is a limited resource, a balance can be struck between the demands for pages versus buffers. If the page cache is too small for the load imposed on the system, the system is constantly swapping pages in and out (moving programs from the swap space into RAM and back again as needed) just to keep up with the current processes. If the page cache is only slightly undersized, the effects are seen not in swapping overhead but in reduced cache performance when running the same programs repeatedly. This is because sufficient pages are available to handle current processes effectively, but there are none to spare for keeping recently used pages in memory for potential access savings.

The **NBUF** parameter specifies the number of buffers in the system buffer cache. **NHBUF** specifies the number of hash queues in the buffer cache. Rather than search the entire pool of buffers, the buffer cache is broken into a series of queues that are organized, or "hashed", by device and block number. The more buffers, the greater chance that data can be found in the buffers without the system having to do a time-consuming disk read. The **sar -b** and **sar -w** commands indicate how effective the system buffers are. (See "Using performance tools to diagnose system inefficiency" later in this chapter.) The value for **NHBUF** must be a power of 2; in addition, **NBUF** divided by **NHBUF** must be approximately 4. (See also **MAXBUF** in the section "Disks and buffers" later in this chapter.)

If you choose to modify the number of buffers after the system has run for a day or so, check system performance, particularly excessive swapping activity. If such activity is found, reduce the number of buffers. By using **sar -w**, you can also determine how many programs are swapped in and out during a given interval. If excessive swapping is evident, you can reduce the number of buffers, which increases the size of the page cache by making more memory available. We also recommend adding as much RAM as practical; swapping is decreased and performance is improved.

Adding memory (RAM)

In the past, administrators of UNIX systems would routinely increase all system tunable parameters when additional memory was installed in minicomputers and superminicomputers. This would usually allow the system to support more users without encountering system limits during heavy system activity. For a single-user PC environment, however, there may be no need to increase kernel tunable parameters at all. And, for the reasons previously stated, keeping system limits at the default value may deliver optimum performance even when additional memory is installed.

The default parameters defined in the "Tunable system parameter descriptions" section (at the end of this chapter) are intended for a system with 8 to 12 Mbytes of RAM. If your system is used in a multiuser configuration (of five users or more, for example) you may wish to add more memory and increase selective parameters to be sure system limits are not reached, and to increase the size of the buffer cache. Table 17.2 gives suggested parameter values for various system configurations. The table shows how, for higher performance systems, the parameter values can be increased to give better system performance. You should not, however, use the values given in the table as a basis for your own system, which may differ considerably from the systems used to gather the test data. Instead, you should try to establish a performance baseline for your system (using **sar** and **timex**), make changes to the parameter values, and again determine your system's performance. This is the best approach to see if parameter changes increase or decrease your system's performance. But, remember, if swapping is occurring, adjusting parameter values will have very little effect on system performance; you should first increase the amount of RAM on your system to reduce, or preferably eliminate, swapping.

Table 17-2 Suggested kernel parameter values

Parameter	Memory 3.5Mb Processor 386/25 Disk space 35Mb	Memory 32Mb Processor 486/33 Disk space 150Mb	System Memory 24Mb* Processors 2x486/33 Disk space 2x204Mb	Memory 56Mb* Processors 4x486/33 Disk space 2x204Mb
NBUF	500	7000	7000	8000
NINODE	200	3000	3000	3000
NFILE	150	1500	3000	3000
NPROC	100	800	1000	1000
NCLIST	120	2000	2048	2048
MAXUP	50	100	100	100
NHINODE	64	2048	2048	2048
NHBUF	256	2048	8192	8192
NMPHEADBUF	150	300	300	300
NPBUF	20	30	30	40
NMPBUF	0	50	50	50
S5CACHEENTS	256	500	500	1024
S5HASHQS	61	499	499	1009
S5OFBIAS	8	14	14	16

NOTE The two systems marked with asterisks (*) are MPX (Multiprocessor Extension) systems. With MPX systems there is an advantage in having **NHBUF** greater than **NBUF** since this reduces the amount of time spent searching for a given buffer, and therefore reduces contention between the processors. With single processor systems, **NBUF** should normally be four times as large as **NHBUF**.

You should be very careful when adjusting parameter values and take note that while the performance of one aspect of the system may be increased the performance of a different aspect may be decreased. Increasing parameter values also means that there is less memory available for user processes, which can also lower overall performance. As a guideline to kernel memory usage, Table 17.3 shows how much memory is used up by increasing the value of key parameters. For example, increasing **NBUF** by 1 uses up 1096 bytes of kernel memory.

Table 17-3 Parameter memory usage

Parameter	Memory Usage
NBUF	72 bytes + 1k each
MAXBUF	72 bytes each
NINODE	76 bytes each
NFILE	12 bytes each
NPROC	344 bytes each
NCLIST	12 bytes each
MAXUP	free
NHINODE	8 bytes each
NHBUF	16 bytes each
NMPHEADBUF	72 bytes each
NPBUF	72 bytes each
NMPBUF	72 bytes each
S5CACHEENTS	72 bytes each
S5HASHQS	8 bytes each
S5OFBIAS	free
NREGION	68 bytes each
MAXUMEM	free

Reorganizing filesystems

As filesystems are used, the blocks of individual member files tend to become physically scattered around the disk(s) and I/O becomes less efficient. This scattering yields poor ordering of blocks within files and poor directory structure. Directories also tend to grow large and increase search time. These problems increase file access overhead and are discussed in "Maintaining efficient filesystem organization" in the "Managing filesystems" chapter in this guide.

Defining efficient system usage patterns

After the kernel and the system activities are tuned, and the filesystems organized, the next step for improving system performance is to perform some housekeeping activities and to check whether prime-time load can be reduced. The person responsible for administering the system should check for the following:

- less important (or even unnecessary) jobs interfering with more important jobs
- the efficiency of user-defined features, such as *.profile* and **$PATH**

Checking process activity with ps

The **ps**(C) command obtains information about active processes. This command gives a "snapshot" picture of what processes are executing, which is useful when you are trying to identify what processes are loading the system. Things will probably change by the time the output appears; however, the entries that you should be interested in are **TIME** (minutes and seconds of CPU time used by processes) and **STIME** (time when process first started). Example 17-1 contains sample output from the **ps -afe** command.

Example 17-1 Sample output from ps -afe

```
    UID   PID  PPID  C    STIME TTY  TIME COMMAND
   root 22247 21299  0 16:54:41  T1  0:01 mscreen -n 2
johnson 22246 22079  0 16:52:53  T0  0:04 vi file2
   root 22285 22247  0 16:56:04  T1  0:06 mscreen -n 2
   root 22284 22247  0 16:56:04  T1  0:00 mscreen -n 2
markham 22274 22271  0 16:55:09  p1  1:05 rlogin colossus
markham 22271 22243  0 16:55:07  p1  1:05 rlogin colossus
 forbin 22304 21003  0 17:02:51 003  0:00 /usr/bin/mail kuprin
 fisher 22298 18505  2 17:02:29 011  3:28 nethack
   root 22305  1327 10 17:02:57  T2  0:00 ps -afe
```

The "Troubleshooting your system" chapter explains how to use the **ps** command to perform such tasks as locating "runaway" processes (one that uses progressively more system resources over a period of time while you are monitoring it). You can also use **ps** to find processes that take a very long time to execute; you can consider using **cron**(C) to execute the such jobs out of office hours. "Managing processes" in the *User's Guide* explains how to do this.

Checking user $PATH variables

The **$PATH** environment variable lists the pathnames of all directories to be searched each time a command is executed. **$PATH** is defined in a user's *.login*, *.cshrc*, or *.profile* file. (See **environ**(M) for more information on environment variables.) Before displaying "not found," the system must search every directory in **$PATH**. These searches require both processor and disk time. If there is a disk or processor bottleneck, changes here can help performance.

Some things that you should check for in user **$PATH** variables are:

- path efficiency

  $PATH is read left to right, so the most likely places to find the command should be first in the path (**/bin** and **/usr/bin**). Make sure that a directory does not appear twice in **$PATH**.

- path length

  In general, **$PATH** should have as few entries as possible.
- large directory searches

  Searches of large directories should be avoided if possible. Put any large directories at the end of **$PATH**.

Using performance tools to diagnose system inefficiency

The operating system includes a series of tools to measure performance. These tools can be used by the system administrator to locate problem areas. The performance tools for the internal activities described in this section are:

sar	(System Activity Reporter) samples the state of the system and provides reports on various system-wide activities.
timex	reports both system-wide and per-process activity during the execution of a command or program.

The sar command

Internal activity is measured by a number of counters contained in the kernel. Each time an operation is performed, an associated counter is incremented. The **sar**(ADM) utility generates reports based on the raw data gathered from these counters. **sar** reports can be used to diagnose system problems. The two most critical areas to monitor are memory and CPU (central processing unit) usage. The functions monitored by **sar** are discussed in the subsections that follow, including analysis of sample **sar** output. **sar** can either gather system activity data "live" or extract information collected in data files created by **sadc** (System Activity Data Collector). By default, the following **crontab** entry is installed in */usr/spool/cron/crontabs/sys*:

```
0 * * * 0-6 /usr/lib/sa/sa1
20,40 8-17 * * 1-5 /usr/lib/sa/sa1
5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200 -A
```

The sa1 entry produces records every 20 minutes during working hours and hourly otherwise. The sa2 entry writes a daily average report into the same file every hour during work hours. You can modify these entries as desired. The output files are in binary format (for compactness) and are stored in */usr/adm/sa*. The filenames have the format sa*dd*, where *dd* is the day of the month. (See the **crontab**(C) and **sar**(ADM) manual pages for more information on modifying the default **crontab** file.)

The basic syntax of the **sar** command is as follows:

```
sar option t
```

where

option	is one of the options described in the following sections.
t	is the sampling interval in seconds, which should be five or greater.

You must supply a sampling interval if you wish to view data in real time; otherwise data already gathered for that day (in */usr/adm/sa*) is displayed. From the */usr/adm/sa* directory, you can examine all the available records. For example, to get a cumulative report on all **sar** data gathered (**-A** option) on the 23rd of the month (filename is *sa23*), you enter the following commands:

```
cd /usr/adm/sa
sar -A -f sa23 | more
```

The examples in this section are not intended to represent benchmarks; they serve to illustrate how the output can be used. When tuning your system, it is recommended that you use a "benchmark" (a program used to evaluate the performance of a system) and have the system under normal load for your application.

Buffer activity: sar -b

The **-b** option reports the following buffer activity.

bread/s	average number of physical blocks read into the system buffers from the disk (or other block devices) per second
lread/s	average number of logical blocks read from system buffers per second
%rcache	fraction of logical reads found in buffer cache (100% minus the ratio of breads to lreads)
bwrit/s	average number of physical blocks written from the system buffers to disk (or other block devices) per second
lwrit/s	average number of logical blocks written to system buffers per second
%wcache	fraction of logical writes found in buffer cache (100% minus the ratio of bwrit/s to lwrit/s)
pread/s	average number of physical read requests per second
pwrit/s	average number of physical write requests per second

The entries that you should be most interested in are the cache hit ratios %rcache and %wcache which measure the effectiveness of system buffering. If %rcache falls below 90, or %wcache falls below 65, it may be possible to improve performance by increasing the number of buffers.

An example of **sar -b** output follows:

```
unix unix 3.2 2 i386    02/18/89
16:32:57 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
16:33:07       3      39      93       1      16      91       0       0
16:33:17       4      40      90       2      16      87       0       0
16:33:27       4      41      90       3       7      64       0       0
Average        4      40      91       2      13      84       0       0
```

This example shows that the buffers are not causing any bottlenecks, because all data is within acceptable limits.

Name cache activity: sar -n

The **sar -n** option reports the name cache statistics.

c_hits	the number of cache hits
cmisses	the number of cache misses
hit%	the hit to miss ratio as a percentage

An example of **sar -n** follows:

```
unix unix 3.2 2 i386    02/18/89

11:26:05  c_hits cmisses (hit %)
11:26:10       9       0 (100%)
```

The name cache is described in detail in "Tunable disk efficiency schemes."

Process throughput: sar -q

The **sar -q** option reports the average queue length while the queue is occupied and percent of time occupied.

runq-sz	run queue of processes in memory; typically, this should be less than 2. Consistently higher values mean you are CPU-bound
%runocc	the percentage of time the run queue is occupied; the larger this value is the better
swpq-sz	swap queue of processes to be swapped out; the smaller this number is the better
%swpocc	the percentage of time the swap queue is occupied; the smaller this value is the better

An example of **sar -q** follows:

```
unix unix 3.2 2 i386    02/18/89
11:00:56 runq-sz %runocc swpq-sz %swpocc
11:01:07     1.7      98     1.5      36
11:01:17     1.0      63     1.0      31
11:01:27     1.0      58     1.0      49
Average      1.3      74     1.2      39
```

In this example, the processor utilization (%runocc) varies between 58% and 98%, while the fraction of time the swap queue is not empty (%swpocc) is 31% to 49%. This means that memory is not causing a major bottleneck in the system throughput, but more memory would help reduce the swapping/paging activity.

If %runocc is greater than 90 and runq-sz is greater than 2, the CPU is heavily loaded and response is degraded. In this case, additional CPU capacity may be required to obtain acceptable system response. If %swpocc is greater than 20, more memory or fewer buffers would help reduce swapping/paging activity.

CPU utilization: sar -u

The CPU utilization is listed by **sar -u** (default). At any given moment the processor is either busy or idle. When busy, the processor is in either user or system mode. When idle, the processor is waiting for input/output completion or has no work to do. The **-u** option of **sar** lists the percent of time that the processor is in system mode (%sys), user mode (%user), waiting for input/output completion (%wio), and idle time (%idle).

In typical timesharing use, %sys and %usr are about the same value. In special applications, either of these can be larger than the other without anything being abnormal. A high %wio generally means a disk bottleneck. A high %idle, with degraded response time, may mean memory constraints; time spent waiting for memory is attributed to %idle.

The "idle" (percentage idle) column can also provide some insight into system performance. This figure is normally between 40 and 100 percent, even with a large number of active users. When this figure falls consistently below 30%, the chief competition for resources does not involve memory at all; the critical resource is raw processor power. (Run the **ps**(C) command to make certain that the excessive CPU usage is not due to a runaway process that is stealing every spare CPU cycle.)

If you are running a large number of users, it may help to switch to smart serial boards if you are using more common dumb cards. Smart cards take some of the burden off the CPU rather than adding to the amount of work it has to do.

In addition, you should examine */usr/spool/crontab* to see if jobs are queued up for peak periods that might better be run at times when the system is idle. Use the **ps** command to determine what processes are heavily loading the system. Encourage users to run large, non-interactive commands (such as **nroff**(CT) or **troff**(CT)) at off-peak hours. You may also want to run such commands with a low priority by using the **nice**(C) or **batch**(C) commands. An example of **sar -u** follows:

```
unix unix 3.2 2 i386    02/18/89
09:20:08 %usr       %sys     %wio     %idle
09:40:12    6          7        2        86
10:00:03    7          9        3        80
10:20:07   14         16       10        61
Average     9         11        5        76
```

System tables: sar -v

The **-v** option reports the status of process, inode, file, shared-memory record, and shared-memory file tables. From this report you know when the system tables need to be modified.

proc-sz	number of process table entries presently being used/allocated in the kernel
inod-sz	number of inode table entries presently being used/allocated in the kernel
file-sz	number of file table entries presently being used/allocated in the kernel
ov	number of times an overflow occurred (one column for each of the above three items)
lock-sz	the number of shared memory record table entries presently being used/allocated in the kernel

The values are given as level/table size, meaning the current number of table entries in use and the size of the table. An example of **sar -v** follows:

```
unix unix 3.2 2 i386     02/18/89
17:36:05 proc-sz ov   inod-sz ov   file-sz ov   lock-sz
17:36:35  17/ 40  0    39/ 80  0    29/ 80  0     0/ 50
17:37:05  19/ 40  0    46/ 80  0    35/ 80  0     0/ 50
17:37:35  18/ 40  0    43/ 80  0    34/ 80  0     0/ 50
```

This example shows that all tables are large enough to have no overflows. Sizes could be reduced to save main memory space if these are the highest values ever recorded.

Swapping activity: sar -w

The **-w** option reports swapping and switching activity. The following are some target values and observations.

swpin/s	number of transfers into memory per second
bswin/s	number of 512-byte-block units (blocks) transferred for swap-ins (including initial loading of some programs) per second
swpot/s	number of transfers from memory to the disk swap area per second. If greater than 1, memory may need to be increased or buffers decreased
bswot/s	number of blocks transferred for swap-outs per second
pswch/s	process switches per second. This should be 30 to 50 on a busy 4 to 6 user system

An example of **sar -w** output follows:

```
unix unix 3.2 2 i386    02/18/89
19:53:44 swpin/s bswin/s swpot/s bswot/s pswch/s
19:53:58     0.0     0.0     0.0     0.0      37
19:54:14     0.0     0.0     0.0     0.0      39
19:54:24     0.0     0.0     0.0     0.0      39
Average      0.0     0.0     0.0     0.0      38
```

This example shows that there is sufficient memory for the currently active users, because no swapping is occurring.

Examining system activity per command: timex

The **timex** command times a command and reports the system activities that occurred during the time the command was executing. If no other programs are running, then **timex** can give you a good idea of which resources a specific command uses during its execution. System consumption can be collected for each application program and used for tuning the heavily loaded resources. **timex** can be used in the following way:

timex -s ***program***

Your application program operates normally. When you finish and exit, the **timex** result is printed on your screen. You can then get a clear picture of system resources used by your program.

Tunable disk efficiency schemes

Your system also includes two schemes that can increase the efficiency of your system:

Name cache	The names of recently used files are stored in a cache. This can reduce the time the system spends connecting a file name with its contents. (Executables and data files are treated equally.) If files are accessed multiple times, this increases the efficiency of your system.
Scatter-gather I/O	This is a scheme wherein groups of disk I/O requests are gathered together rather than executed singly. This process increases overall disk efficiency.

This section explains how to tune the parameters associated with these schemes to improve hard disk I/O performance.

Name cache

Name caching involves three tunable parameters:

S5CACHEENTS	number of name components in name cache
S5HASHQS	number of hash queues for name cache
S5OFBIAS	bias towards keeping the names of open files in cache

These three parameters are set to provide roughly a 95% cache hit rate (meaning 95% of requests for a given file are satisfied by looking in the name cache rather than going to the disk or the buffer cache). The default values should suffice for most multiuser tasks, such as a group of users running a small suite of applications. They should also prove sufficient for occasional single-user tasks such as a large source code compile. If your system is achieving less than a 90% cache hit rate, the caching parameters should be increased. (The **sar -n** provides data on the name cache hit rate. Refer to the "Using performance tools to diagnose system inefficiency" section of this chapter.)

The basic formula for large diverse workgroups is to make **S5CACHEENTS** large (roughly **NINODE*3**), make **S5HASHQS** a prime number roughly 1/4 the size of **S5CACHEENTS**, and leave **S5OFBIAS** alone unless the relevant application packages have various parts that open the same files. For example, the C compiler generates temporary files that are written by one pass of the compiler and read by another. An environment where compilation is a major activity might benefit from increasing **S5OFBIAS**.

A value of 0 for **S5OFBIAS** means that the names of open files have no special caching priority, and are as likely to be dropped from the cache as any other cached item. If **S5OFBIAS** is equal to **S5CACHEENTS**, open files almost always get to keep their names in the cache. **S5OFBIAS** should not be greater than **S5CACHEENTS/10** unless **S5CACHEENTS** is quite large (at least several times greater than **NINODE**). A combination of a **S5CACHEENTS** less than twice **NINODE** and an **S5OFBIAS** greater than **S5CACHEENTS/10** can cause very poor name-lookup performance, possibly even worse than not having the name cache in place at all.

Scatter-gather I/O

Scatter-gather I/O groups disk requests together instead of executing them one at a time.

NOTE The scatter-gather feature introduced here is not to be confused with support for SCSI scatter-gather, a hardware-based feature exclusive to certain SCSI host adapters. The former is a software-based implementation that uses a series of kernel parameters to set and adjust this functionality.

The Acer fast filesystem uses a tunable parameter **NMPBUF** that controls the number of buffers used to gather large disk requests before transferring the contents to user space or the buffer cache. The **NMPBUF** buffers are expensive in terms of memory consumption (approximately 16K each). Another parameter, **NMPHEADBUF**, can be used to allocate standalone cluster buffer headers, which are essentially **NMPBUF** buffer headers without their corresponding 16K buffer. These are relatively cheap and can increase disk performance for certain configurations.

Configuring fast filesystem buffers is a tradeoff of memory usage and speed. The recommendations that follow allow you to reduce memory requirements and still retain full performance. If memory conservation is not an issue on your system, setting **NMPBUF** to 16 and **NMPHEADBUF** to **NBUF/3** should be sufficient for most configurations. The value of **NBUF** is determined at boot time and displayed on the console as "i/o bufs." (This information can also be obtained by looking in the file */usr/adm/messages*.) The boot display looks like this:

```
kernel: drivers = 180k, 4 screens = 68k, 600 i/o bufs = 600k, msg bufs = 8k
```

ESDI disks and other disks supporting a 1:1 interleave

If you are using a disk with a 1:1 interleave, set **NMPBUF** to 3, and set **NMPHEADBUF** to around 1/3 the number of disk buffers (**NBUF**).

SCSI disks and others unable to use 1:1 interleave efficiently

If you are using a SCSI disk, or MFM/RLL disks that cannot use a 1:1 interleave efficiently, a much lower value of **NMPHEADBUF** should suffice (such as 16), and it saves a little bit of memory. Leave **NMPBUF** at 0 so this value is autoconfigured at boot time.

Tunable system parameter descriptions

The following sections provide a breakdown of system tunable parameters defined in the file */etc/conf/cf.d/mtune*. The parameter categories are as follows:

Disks and Buffers
Character Buffers
Files, Inodes and Filesystems
Processes, Memory Management and Swapping
Clock
MultiScreens
Message Queues
Semaphores
Shared Data
System Name
Streams Data
Event Queues and Devices
Hardware Dependent Parameters
Security
Asynchronous I/O Parameters

NOTE The parameters are listed by category and in the order they are displayed by **configure** instead of alphabetically.

Disks and buffers

NDISK	is the number of disk drives attached to the system. This is set to 4 by default and is increased by **mkdev hd** when additional SCSI disks are installed.
NBUF	is the number of 1K system buffers allocated at boot time. If set to 0, this value is calculated by the system. (**NBUF** is displayed as "kernel i/o/bufs" at boot time; see the file */usr/adm/messages*). The buffers form a data cache. The data cache is a memory array containing disk file information. Cache hit rate increases with the number of buffers. Cache hits reduce the number of disk accesses and thus may improve overall performance. The entries are normally in the range of 100 to 600. Each buffer contains 1076 bytes. The number of hash queues (**NHBUF**) should be increased along with system buffers (**NBUF**) for optimal performance.

NPBUF specifies how many physical I/O buffers to allocate. One I/O buffer is needed for each physical read or write which is active. Each entry contains 52 bytes. The default value is 20.

NHBUF specifies how many hash queues to allocate for 1K buffers. These are used to search for a buffer given a device number and block number rather than a linear search through the entire list of buffers. This value must be a power of 2. Each entry contains 12 bytes. The **NHBUF** value should be chosen so that the value **NBUF** divided by **NHBUF** is approximately equal to 4. The default value is 256.

CTBUFSIZE is the size of the tape buffer in Kbytes. This should have a value of 32 to 256. It is the size of a static buffer allocated at init time. The following are reasonable values for associated circumstances:

32K	bare minimum: insufficient to stream
64K	minimum to allow streaming (good for systems with little memory) or little tape use (performance is not critical)
96K	a first reduction, if default uses too much memory
128K	default: good tradeoff performance
192K	a first increase, if default provides poor performance
256K	maximum

MAXBUF is the maximum possible number of buffers in the buffer cache. This is the number of buffer description headers in the kernel. Fewer than this number of buffers may actually be autoconfigured by the kernel at boot time, depending on how much core is present. If **NBUF** is non-zero, then exactly **NBUF** buffers are configured, and there is no reason for **MAXBUF** to be larger than **NBUF**. If **NBUF** is 0, the kernel configures at most **MAXBUF** buffers automatically. The default value is 600.

DMAABLEBUF is the number of transfer buffers for DMA requests over 16 Mbytes. This must be a value of 4 to 128. The default is 16.

PLOWBUFS	is the amount of buffer cache that is contained in Direct Access Memory (the first 16Mbytes of RAM). It is expressed as a percentage and should be as high as possible, if you do not have a 32-bit controller (indicated by `fts=d` on the boot screen), to limit the number of copying requests made from buffers above 16Mbyte (see **NCOPYBUFS**). The default value is 100%. This parameter need only be changed if you have more than 16Mbytes of RAM on your system. If you have 16Mbytes (or less) of RAM you have no choice but to have all your buffer cache in Direct Access Memory.
NCOPYBUFS	is the number of buffers available for copying requests made from buffers above 16Mbyte to buffers below 16Mbyte, for non 32-bit controllers.
NAUTOUP	specifies the buffer age in seconds for automatic filesystem updates. This parameter has a default value of 10. A system buffer is written to the hard disk when it has been memory-resident for the interval specified by the **NAUTOUP** parameter. Specifying a smaller limit increases system reliability by writing the buffers to disk more frequently and decreases system performance. Specifying a larger limit increases system performance at the expense of reliability. This parameter controls behavior of the **bdflush** daemon process.
BDFLUSHR	specifies the rate in seconds for checking the need to write the filesystem buffers to the disk. The range is 1 to 300. The default is 30 seconds. This parameter controls behavior of the **bdflush** daemon process.
PUTBUFSZ	specifies the size of a circular buffer, **putbuf**, that contains a copy of the last **PUTBUFSZ** characters written to the console by the operating system. The default is 2000. The contents of **putbuf** can be viewed using **crash**(ADM).
PIOMAP	determines the size of the map entry array used by the kernel programmed I/O (PIO) breakup routine. This routine allows device drivers to do programmed I/O of large data blocks at interrupt level by breaking the data blocks into smaller data units. Users should not modify this parameter.
PIOMAXSZ	is the maximum number of pages to use at one time for programmed I/O. Users should not modify this parameter.

Character buffers

The following parameters control various data structure sizes and other limits in base system device drivers.

NCLIST	specifies how many character list buffers to allocate. Each buffer contains up to 64 bytes. The buffers are dynamically linked to form input and output queues for the terminal lines and other slow-speed devices. The average number of buffers needed per terminal is in the range of 5 to 10. Each entry (buffer space plus header) contains 72 bytes. When full, input and output characters dealing with terminals are lost, although echoing continues. The default value is 120.
NEMAP	specifies the maximum number of I/O translation mappings.
NUMXT	determines the number of layers a subdevice can configure to support bitmapped display devices such as the BLIT or the AT&T 5620 terminal.
NUMSXT	determines the number of shell layers a subdevice can configure. This has a default value of 6.
NKDVTTY	determines the number of virtual terminals (ttys) supported by the console keyboard driver. Users should not modify this parameter.
NCPYRIGHT	defines the size of a kernel data structure used to print console initialization messages. Users should not modify this parameter.
MAX_CFGSIZE	is the maximum size of configuration information saved by the Streams driver. The default value is 1024.
PRFMAX	is the maximum number of text symbols that the kernel profiler (*/dev/prf*, described in **profiler**(ADM)) can properly process.
KDBSYMSIZE	is the size of the symbol table. Must have a value of 50000 to 500000. The default is 300000.

Files, inodes, and filesystems

NINODE	specifies how many inode table entries to allocate. Each table entry represents an in-core inode that is an active file. For example, an active file might be a current directory, an open file, or a mount point. The file control structure is modified when changing this variable. The number of entries used depends on the number of opened files. The entries are normally in the range of 100 to 400. The default is 300. The value for **NINODE** pertains directly to the **NFILE** value. (**NINODE** is equal to or greater than **NFILE**). When the i-node table overflows, the following warning message is displayed on the system console: `WARNING: inode table overflow`
NHINODE	specifies the size of the inode hash table. The default value is 128.
NFILE	specifies how many open file table entries to allocate. Each entry represents an open file. The entry is normally in the range of 100 to 600. The default is 200. Each entry contains 12 bytes. The **NFILE** entry relates directly to the **NINODE** entry. (**NFILE** is less than or equal to **NINODE**.) The **NFILE** control structure operates in the same manner as the **NINODE** structure. When the file table overflows, the following warning message is displayed on the system console: `NOTICE: file table overflow` As a reminder, this parameter does not affect the number of open files per process (see the **NOFILES** parameter).
NMOUNT	specifies how many mount table entries to allocate. Each entry represents a mounted filesystem. The root (/) filesystem is always the first entry. When full, the **mount** system call returns the error `EBUSY`. Because the mount table is searched linearly, this value should be as low as possible. The default value is 8.
CMASK	is the default mask **umask**(S) used for file creation. By default, this is zero, meaning that the **umask** is not set in the kernel.

ETRUNC truncates newly created filenames silently when set to 1. On S51K, XENIX, and AFS filesystems, filenames are truncated to 14 characters. On EAFS and ES51K filesystems, filenames are truncated to 255 characters. When left at default of 0, attempts to create files with names greater than these respective limits fail with **ENAMETOOLONG**, which is the behavior mandated by POSIX FIPS requirements.

Table 17-4 Filename truncation

	TRUNC 0	TRUNC 1
S51K	Fails	Truncates to 14
XENIX	Fails	Truncates to 14
AFS	Fails	Truncates to 14
EAFS	Fails	Truncates to 255
ES51K	Fails	Truncates to 255

NOFILES specifies the maximum number of open files per process. The default is 60. Unless an application package recommends that **NOFILES** be changed, the default setting of 60 should be left as is. */bin/sh* uses three file table entries: standard input, standard output, and standard error (0, 1, and 2 are normally reserved for stdin, stdout, and stderr, respectively). This leaves the value of **NOFILES** minus 3 as the number of other open files available per process. If a process requires up to three more than this number, then the standard files must be closed. This practice is not recommended and must be used with caution, if at all. If the configured value of **NOFILES** is greater than the maximum (100) or less than the minimum (60), the configured value is set to the default (60), and a message is sent to the console.

SHLBMAX specifies the maximum number of shared libraries that can be attached to a process at one time. The default value is 8, with a maximum of 16.

FLCKREC	specifies the number of records that can be locked by the system. The default value is 100. Each entry contains 28 bytes.
NMPBUF	is the number of Acer Fast File System cluster buffers. They are used to gather large disk requests before transferring the contents either to user space or to the buffer cache. mpbufs are large (16K each). The value **NMPBUF** should be 0 to 16, the latter being only appropriate for a machine with 16 Mbytes memory or more and many users. When set to zero (the system default) the real value for this parameter is determined at boot time.
NMPHEADBUF	is the number of Acer Fast Filesystem standalone cluster buffer headers. This parameter must have a value from 0 to 600. The default is 150.
BFREEMIN	is the number of buffers that must be on the freelist before a buffer is waited for. It should usually be 0. Machines that are used primarily for media copying, **uucp** transfers, and other applications that are both quasi-single-user and access lots of files see a performance boost by setting this to an appropriate value of around **NBUF**/10. The maximum value is 100.
S5CACHEENTS	is the number of name components in the filename cache. Must have a value of 1 to 1024. Default is 256. The recommended value for diverse workgroups is to make **S5CACHEENTS** large, roughly three times the value of **NINODE**.
S5HASHQS	is the number of hash queues for name cache (best if prime number). Must have a value of 1 to 1021. The default is 61. The recommended value for diverse workgroups is to make **S5HASHQS** a prime number roughly a quarter the size of **S5CACHEENTS**.
S5OFBIAS	determines bias towards keeping open files around in cache. Must have a value of 0 to 256. The default is 8. A value of zero means that the names of open files have no special caching priority. If **S5OFBIAS** is equal to **S5CACHEENTS**, the names of open files almost always remain in the cache.
NGROUPS	is the maximum number of supplemental groups. This value has a default of 8 and a maximum of 16.

Processes, memory management and swapping

A paging daemon, **vhand**, is responsible for freeing up memory as the need arises. It uses a "least recently used" algorithm to approximate process working sets, and it writes out those pages that were not modified during some period of time to the disk. The page size is 4096 bytes. When memory is exceptionally tight, the working sets of entire processes may be swapped out.

The following tunable parameters determine how often the **vhand** and **bmapflush** processes run and under what conditions. The default values should be adequate for most applications.

NPROC specifies how many process table entries to allocate. Each table entry represents an active process. The swapper is always the first entry, and */etc/init* is always the second entry. The number of entries depends on the number of terminal lines available and the number of processes spawned by each user. The average number of processes per user is in the range of 2 to 5 (also see **MAXUP**, default value 25). When full, the **fork**(S) system call returns the error `EAGAIN`. The **NPROC** entry is in the range of 50 to 3000. The default is 100.

MAXUP specifies how many concurrent user processes that a non-super user is allowed to run. The entry is normally in the range of 15 to 60, with a maximum value of 300. This value should not exceed the value of **NPROC** (**NPROC** should be at least 10% more than **MAXUP**). This value is per user identification number, not per terminal. For example, if 12 people are logged in on the same user identification, the default limit would be reached very quickly.

NREGION specifies how many region table entries to allocate. Each **NREGION** entry contains 36 bytes. Most processes have three regions: text, data, and stack. Additional regions are needed for each shared memory segment and shared library (text and data) attached. However, the region table entry for the text of a "shared text" program is shared by all processes executing that program. Each shared-memory segment attached to one or more processes uses another region table entry. This parameter has a range of 100-10000, with a default of 300. If the system runs out of region table entries, the following message is displayed on the system console:

```
Region table overflow
```

MAXPMEM specifies the maximum number of physical pages. The default value of 0 specifies that all available physical memory be used.

ULIMIT	specifies in 512-byte blocks the size of the largest file. that an ordinary user may write. The default value is 2097152; that is, the largest file an ordinary user may write is one gigabyte. The super user may write a file as large as the filesystem can hold. The **ULIMIT** parameter does not apply to reads: any user may read a file of any size.
SPTMAP	determines the size of the map entry array used for managing kernel virtual address space. Users should not modify this parameter.
AGEINTERVAL	specifies the number of clock ticks a process runs before its pages are aged. The default value is 31.
GPGSLO	specifies the low water mark of free memory in pages for **vhand** to start stealing pages from processes. The default is 25. Increase the value to make the daemon more active; decrease the value to make the daemon less active (must be an integer ≥ 0 and < **GPGSHI**).
GPGSHI	specifies the high-water mark of free memory in pages for **vhand** to stop stealing pages from processes. The default is 40. Increase the value to make the daemon more active; decrease the value to make the daemon less active. (The value must be an integer > 0, > **GPGSLO**, and < 25 percent of the number of pages of available memory.)
GPGSMSK	is the mask used by the paging daemon. The default is 0x00000420(hex). Note that this appears as a decimal value in the system. This value should not be changed.
MAXSC	specifies the maximum number of pages that are swapped out in a single operation. The default value is 1.
MAXFC	specifies the maximum number of pages that is added to the free list in a single operation. The default value is 1.
MAXUMEM	specifies the maximum size of a user's virtual address space in pages. This value cannot be greater than 8192. The default is 2560.
MINARMEM	specifies the minimum number of memory pages reserved for the text and data segments of user processes.
MINASMEM	is the threshold value that specifies the number of memory and swap pages reserved for system purposes (unavailable for the text and data segments of user processes).
MINHIDUSTK	specifies the minimum data relocation value such that the user stack and data can share a page table. The **MINHIDUSTK** and **MINUSTGAP** values should not be changed.

MINUSTKGAP see **MINHIDUSTK.**

MAXSLICE specifies in clock ticks the maximum time slice for user processes. After a process executes for its allocated time slice, that process is suspended. The operating system then dispatches the highest priority process and allocates to it **MAXSLICE** clock ticks. **MAXSLICE** must be a value from 25-100. The default is 100.

Clock

NCALL specifies how many call-out table entries to allocate. Each entry represents a function to be invoked at a later time by the clock handler portion of the kernel. This value must be in the range of 30 to 500. The default value is 30. Each entry contains 16 bytes.

Software drivers may use call entries to check hardware device status. When the call-out table overflows, the system crashes and displays the following message on the system console:

```
PANIC: Timeout table overflow
```

TIMEZONE specifies the **timezone** setting referred to in the **ctime**(S) system call. Note that the timezone value is a system default timezone and not the value of the **TZ** environment variable. This parameter can have a value from 0-1440.

DSTFLAG specifies the **dstflag** described for the **ctime**(S) system call. A value of 1 indicates Daylight Savings Time.

MultiScreens

TBLNK controls the console screen saver feature. It is the number of seconds before the screen blanks to save wear on the monitor. **TBLNK** can have a value of 0 to 32767, with zero disabling screen blanks. The default is 0.

NSCRN specifies the number of multiscreens. A value of 0 configures this value at boot time based on the amount of memory installed. The maximum value is 12.

NSPTTYS is the number of pseudo-ttys on the system. The default is 16 and the maximum is 32.

SCRNMEM	is the number of 1024-byte blocks for console screen saves. A value of 0 configures this value at boot time based on the amount of memory installed. The maximum value is 128.

Message queues

The following tunable parameters are associated with interprocess communication messages:

MSGMAP	specifies the size of the control map used to manage message segments. The default value is 100. Each entry contains 8 bytes.
MSGMAX	specifies the maximum size of a message. The default value is 2048. Although the maximum possible size the kernel can process is 64 Kbytes -1, the limit is 8192.
MSGMNB	specifies the maximum length of a message queue. The default value is 4096.
MSGMNI	specifies the maximum number of message queues system-wide (id structure). The default value is 50.
MSGTQL	specifies the number of message headers in the system and, thus, the number of outstanding messages. The default value is 40. Each entry contains 12 bytes.
MSGSSZ	specifies the size, in bytes, of a message segment. Messages consist of a contiguous set of message segments large enough to fit the text. The default value is 8. The value of **MSGSSZ** times the value of **MSGSEG** must be less than or equal to 131,072 bytes (128 Kbytes).
MSGSEG	specifies the number of message segments in the system. The default value is 1024. The value of **MSGSSZ** times the value of **MSGSEG** must be less than or equal to 131,072 bytes (128 Kbytes). Default value is set at boot time.

Semaphores

The following tunable parameters are associated with interprocess communication semaphores:

SEMMAP	specifies the size of the control map used to manage semaphore sets. The default value is 10. Each entry contains 8 bytes.
SEMMNI	specifies the number of semaphore identifiers in the kernel. This is the number of unique semaphore sets that can be active at any given time. The default value is 10. Each entry contains 32 bytes.

SEMMNU	specifies the number of undo structures in the system. The default value is 30. The size is equal to 8*(**SEMUME** + 2) bytes.
SEMMSL	specifies the maximum number of semaphores per semaphore identifier. The default value is 25.
SEMOPM	specifies the maximum number of semaphore operations that can be executed per **semop** system call. The default value is 10. Each entry contains 8 bytes.
SEMUME	specifies the maximum number of undo entries per undo structure. The default value is 10. The size is equal to 8***SEMMNU** bytes.
SEMVMX	specifies the maximum value a semaphore can have. The default value is 32767, which is the maximum value for this parameter.
SEMAEM	specifies the adjustment on exit for maximum value, alias **semadj**. This value is used when a semaphore value becomes greater than or equal to the absolute value of **semop**, unless the program has set its own value. The default value is 16384. The default value is the maximum value for this parameter.
SEMMNS	specifies the number of semaphores in the system. The default value is 60. Each entry contains 8 bytes.
XSEMMAX	specifies the maximum number of XENIX special semaphores allowed systemwide. The minimum value for **XSEMMAX** is 20, the maximum value is 60, and the default value is 60.

Shared data

The following tunable parameters are associated with interprocess communication shared memory:

SHMMAX	specifies the maximum shared-memory segment size. The default value is 524288.
SHMMIN	specifies the minimum shared-memory segment size. The default value is 1.
SHMMNI	specifies the maximum number of shared-memory identifiers systemwide. The default value is 100. Each entry contains 52 bytes.
SHMSEG	specifies the number of attached shared-memory segments per process. The default value is 6. The maximum value is 15.

SHMALL	specifies the maximum number of in-use shared-memory text segments. The default value is 512.
XSDSEGS	specifies the maximum number of XENIX special shared-data segments allowed system wide. The minimum value for **XSDSEGS** is 1, the maximum value is 150, and the default value is 25.
XSDSLOTS	specifies the number of slots per XENIX shared data segment. The maximum number of XENIX special shared data segment attachments system wide is **XSDSEGS*XSDSLOTS**. The minimum value for **XSDSLOTS** is 1, the maximum value is 10, and the default value is 3.

System name

NODE	specifies the system name.

Streams data

The following tunable parameters are associated with STREAMS processing.

NQUEUE	is the number of Streams queues to be configured. Queues are always allocated in pairs, so this number should be even. A minimal Stream contains four queues (two for the Stream head, two for the driver). Each module pushed on a Stream requires an additional two queues. A typical configuration value is 4***NSTREAM**.
NSTREAM	is the number of Stream-head (stdata) structures to be configured. One is needed for each Stream opened, including both Streams currently open from user processes and Streams linked under multiplexers. The recommended configuration value is highly application-dependent, but a value of 32 to 40 usually suffices on a computer for running a single transport provider with moderate traffic.
NBLK*n*	**NBLK4** through **NBLK4096** control the number of Streams data blocks and buffers to be allocated for each size class. Message block headers are also allocated based on these numbers: the number of message blocks is 1.25 times the total of all data block allocations. This provides a message block for each data block, plus some extras for duplicating messages (kernel functions **dupb()**, **dupmsg()**). The optimal configuration depends on both the amount of primary memory available and the intended application.

NMUXLINK	is the maximum number of multiplexer links to be configured. One link structure is required for each active multiplexer link (**STREAMS I_LINK ioctl**). This number is application-dependent; the default allocation of 87 guarantees availability of links.
NSTRPUSH	is the maximum number of modules that may be pushed onto a Stream. This prevents an errant user process from consuming all of the available queues on a single Stream. By default this value is 9, but in practice, existing applications have pushed at most four modules on a Stream.
NSTREVENT	is the initial number of Stream event cells to be configured. Stream event cells are used for recording process-specific information in the **poll** system call. They are also used in the implementation of the Streams **I_SETSIG ioctl** and in the kernel **bufcall**() mechanism. A rough minimum value to configure would be the expected number of processes to be simultaneously using **poll** times the expected number of Streams being polled per process, plus the expected number of processes expected to be using Streams concurrently. The default is 256. Note that this number is not necessarily a hard upper limit on the number of event cells that are available on the system (see **MAXSEPGCNT**).
MAXSEPGCNT	is the number of additional pages of memory that can be dynamically allocated for event cells. If this value is 0, only the allocation defined by **NSTREVENT** is available for use. If the value is not 0 and if the kernel runs out of event cells, it will under some circumstances attempt to allocate an extra page of memory from which new event cells can be created. **MAXSEPGCNT** places a limit on the number of pages that can be allocated for this purpose. Once a page is allocated for event cells, however, it cannot be recovered later for use elsewhere. The default value is 1.
STRMSGSZ	is the maximum allowable size of the data portion of any Streams message. This should usually be set just large enough to accommodate the maximum packet size restrictions of the configured Streams modules. If it is larger than necessary, a single **write** or **putmsg** can consume an inordinate number of message blocks. The default value of 4096 is sufficient for existing applications.

STRCTLSZ	is the maximum allowable size of the control portion of any STREAMS message. The control portion of a **putmsg** message is not subject to the constraints of the minimum/maximum packet size, so the value entered here is the only way of providing a limit for the control part of a message. The default value of 1024 is more than sufficient for existing applications.
STRLOFRAC	is the percentage of data blocks of a given class at which low-priority block allocation requests are automatically failed. For example, if **STRLOFRAC** is 40 and there are forty-eight 256-byte blocks, a low-priority allocation request fails when more than nineteen 256-byte blocks are already allocated. The parameter helps prevent deadlock situations by starving out low-priority activity. The default value of 80 works well for most applications. **STRLOFRAC** must be greater than or equal to 0 and less than or equal to **STRMEDFRAC**.
STRMEDFRAC	is the percentage cutoff at which medium priority block allocations are failed (see **STRLOFRAC**). The default value of 90 works well for most applications. **STRMEDFRAC** must be greater than or equal to **STRLOFRAC** and less than or equal to 100. (There is no cutoff fraction for high-priority allocation requests; it is effectively 100.)
NLOG	is the number of minor devices to be configured for the log driver; the active minor devices are 0 through (**NLOG-1**). The recommended value of 3 services an error logger (**strerr**) and a trace command (**strace**), with one left over for miscellaneous usage. If only an error logger and a tracer are to be supported, this number can be set to 2. If there are several daemons for an application that may be submitting log messages, this number can be increased to accommodate the extra users.
NUMSP	determines the number of Streams pipe devices (*/dev/sp*) supported by the system. Users should not modify this parameter.
NUMTIM	is the maximum number of Streams modules that can be pushed by the Transport Library Interface (TLI). This value controls the number of data structures used to hold pushed Streams modules configuration data. Users should not modify this parameter.
NUMTRW	is the number of Transport Library Interface (TLI) read/write data structures to allocate in kernel data space. Users should not modify this parameter.

Event queues and devices

EVQUEUES	is the maximum number of open event queues systemwide. The acceptable range is 1-16, with a default value of 8.
EVDEVS	is the maximum number of devices attached to event queues systemwide. The acceptable range is 1-16, with a default value of 16.
EVDEVSPERQ	is the maximum number of devices per event queue. The acceptable range is 1-16, with a default value of 3.

Hardware dependent parameters

DMAEXCL	specifies whether simultaneous DMA requests are allowed. Some computers have DMA chips that malfunction when more than one allocated channel is used simultaneously. **DMAEXCL** is set to 0 by default to allow simultaneous DMA on multiple channels. If this causes a problem, set it to 1.
KBTYPE	is set to 0 for XT-type keyboards and 1 for AT-type keyboards.
VGA_PLASMA	is set to 1 if a VGA gas plasma display is present, 0 if not.
NSHINTR	is the maximum number of devices sharing the same interrupt vector. This has a default value of 8. Users should not modify this parameter.
DO387CR3	controls the setting of high-order bits of Control Register 3 (CR3) when an 80387 math coprocessor is installed. The default value is 0 (switched off).

NOTE The following two parameters are available via **mtune**(F), but not **configure**(ADM).

NAHACCB	is the number of mailboxes available for the adaptec driver to talk to the adaptec hardware. The higher the number, the less likely it is that the driver has to "sleep" before "talking" to the hardware. It is not normally necessary to modify this parameter.
SDSKOUT	is the number of outstanding requests on a SCSI disk at any one time.

Security

There are three parameters that are defined according to the security scheme in place. They can also be configured individually as desired. These parameters are also discussed in the "Administering user accounts" and "Maintaining system security" chapters.

SECLUID	controls the enforcement of LUID (login user ID). This is set to 1 (ON) on a system running in High security mode and 0 (OFF) in all other modes.
SECSTOPIO	controls the usage of **stopio**(S) calls to ensure a device is not held open by another process after it is allocated to another user. This is set to 1 (ON) on a system running in High security mode and 0 (OFF) in all other modes.
SECCLEARID	controls the clearing of SUID/SGID bits when a file is written. This is set to 1 (ON) on a system running in High security mode and 0 (OFF) in all other modes.

Asynchronous I/O

NAIOPROC	is the number of processes which may be simultaneously performing asynchronous I/O. The default value is 5.
NAIOREQ	is the maximum number of pending asynchronous I/O requests. The default value is 120.
NAIOBUF	is the number of asynchronous I/O buffers. This should always be set to the same value as **NAIOREQ**.
NAIOHBUF	is the number of asynchronous hash queues (internal).
NAIOREQPP	is the maximum number of asynchronous I/O requests that a single process can have pending. The default value is 120, meaning that a single process can potentially exhaust all asynchronous I/O resources.
NAIOLOCKTBL	is the number of entries in the internal kernel table for asynchronous **I/O** lock permissions. The default value is 10. If there are many entries in the */usr/lib/aiomemlock* file, this value may need to be increased.

Boot load extension parameters

NOTE This group of parameters is available via **mtune**(F), but not **configure**(ADM).

EXTRA_NDEV	is the number of extra device slots in **fmodsw[ ]**, **io_init[ ]**, and **io...[ ]**. It defines the number of slots reserved in the device driver tables for Boot Time Loadable Drivers.
EXTRA_NEVENT	is the number of extra event slots. It defines the number of slots reserved in the event driver tables for Boot Time Loadable Drivers.
EXTRA_NFILSYS	is the number of extra types of filesystem. It defines the number of extra types of filesystem that can be mounted at boot time.
MAX_BDEV	is the maximum number of block devices (**bdevcnt** is at least this value). It defines the minimum number of entries in **bdevsw[ ]**, the block device switch table.
MAX_CDEV	is the maximum number of character devices (**cdevcnt** is at least this value). It defines the minimum number of entries in **cdevsw[ ]**, the character device switch table.

Multiprocessing parameters

MAXACPUS	is the number of additional CPUs. You should not modify this parameter.

Chapter 18 Building a remote network with UUCP

This chapter explains how to use UUCP to build a remote network system for your computer using a normal telephone line and a modem.

NOTE UUCP is not a terminal emulation program. If you want to use your modem to dial into another computer and log on, you should refer to the "Using modems" chapter of this guide and follow the instructions for adding dial-in and dial-out modems.

If you plan to do extensive file transfers between physically separated XENIX and UNIX systems, you should set up a UUCP connection.

What is UUCP?

The UUCP package permits XENIX and UNIX systems to communicate as part of a remote network. The name UUCP is an acronym for "UNIX-to-UNIX Copy". The UUCP package consists of a group of programs that provide the following capabilities:

- remote file transfer (**uucp**)
- remote command execution (**uux**)
- mail to and from remote sites (via **mail**)

The UNIX system uses the HoneyDanBer implementation of UUCP. Used primarily over phone lines, UUCP can connect with specific remote machines on a demand or scheduled basis, and by either dialing out or allowing other machines to call in.

UUCP uses a batch method to manage communications traffic, storing (or "spooling") requests for later execution when actual contact is made between systems. When UUCP commands are executed, work files and any data files needed are created in */usr/spool/uucp* and its subdirectories. The program **uucico** scans these directories for the instructions contained in any work files and executes them. Although it is possible to execute commands immediately, most systems call other systems according to a daily schedule (usually during the evenings to reduce connection costs).

How to use this chapter

This chapter describes how to build a UUCP system and covers both hardware installation and software configuration. There are also sections on routine maintenance and troubleshooting.

The following is a procedural outline of what must be done to set up your UUCP network:

1. Connect and configure a modem or direct wire.
2. Configure the UUCP software using **uuinstall.**
3. Create login accounts for any sites that will be calling your system.
4. Test your connections with each remote site.

NOTE If you are planning to route mail over your UUCP system, see the chapter "Setting up electronic mail" in this guide for instructions on configuring mail traffic to work over UUCP.

The most important task of configuring UUCP is the editing of several control files that act as the database for UUCP. The next few sections describe the function of these files, and "Configuring UUCP on your system" later in this chapter explains the information that these files contain. The **uuinstall** utility edits these files for you and explains each entry; **uuinstall** also includes an extensive help facility. Read "Configuring UUCP on your system" carefully before running **uuinstall** so that you understand the UUCP database.

What you need

To set up your UUCP communication system, you need:

- at least one RS-232 serial line (or serial port) on your computer to use for UUCP
- the UUCP and MAIL packages extracted from your UNIX system distribution using **custom**(ADM)
- a modem. Supported modems include models by Hayes, Penril, Ventel, Vadic, Rixon, AT&T, and Telebit. You can supply *Dialers* entries or dialer programs for other modems. (For best results, use dialer programs.)
- a standard telephone jack for access to the telephone system
- a cable to connect the serial port to the modem

UUCP commands

UUCP programs are divided into two categories: user programs and administrative programs. The paragraphs that follow describe the programs in each category.

User programs

The user programs for basic networking are in */usr/bin*. No special permission is needed to use these programs, although it is possible to restrict access to the devices they control. These commands are all described in the "Communicating with other sites" chapter of the *User's Guide*.

cu	connects your computer to a remote computer so you can be logged in on both at the same time. You can transfer files or execute commands on either computer without dropping the initial link.
ct	connects your computer to a remote terminal so the user of the remote terminal can log in. The user of a remote terminal can call the computer and request that the computer call it back. The computer then drops the initial link so that the remote terminal's modem is available when it is called back.
uucp	copies files from one computer to another. It creates work files and data files, queues the job for transfer, and calls the **uucico** daemon, which contacts the remote computer.
uupick	retrieves the files placed in */usr/spool/uucppublic/receive* when files are transferred using **uuto**.

uustat	displays the status of requested transfers (**uucp, uuto,** or **uux**). It also provides a means of controlling queued transfers.
uuto	copies files from one computer to a public spool directory on another computer in */usr/spool/uucppublic/receive*. Unlike **uucp,** which lets you copy a file to any accessible directory on the remote computer, **uuto** places the file in an appropriate spool directory, and tells the remote user to pick it up with **uupick**.
uux	creates the work, data, and execute files needed to execute commands on a remote computer. The work file contains the same information as work files created by **uucp** and **uuto**. The execute files contain the command string to be executed on the remote computer and a list of the data files. The data files are those files required for the command execution.

Administrative programs

Most of the administrative programs, control files, and scripts are in */usr/lib/uucp*. Two exceptions are **uuinstall** and **uulog**, which are in */etc* and */usr/bin*, respectively.

uucheck	checks for the presence of basic networking directories, programs, and support files. It also checks the *Permissions, Systems*, and *Devices* files for syntax errors.
uuclean	cleans up the spool directory. It is normally executed from a shell script called **uudemon.clean**, which can be set up to be run by **cron**.
uulog	displays the contents of a specified computer's log files. Log files are created for each remote computer your computer communicates with. The log files contain records of each use of **uucp, uuto,** and **uux**.
uutry	tests call-processing capabilities and does a moderate amount of debugging. It invokes the **uucico** daemon to establish the communications link.

UUCP directories

There are three directories associated with UUCP:

/usr/spool/uucp	This is the working directory for UUCP. Work files, lock files, log files, and all UUCP communications traffic are stored here and in subdirectories.
/usr/spool/uucppublic	This is the publically readable or writable target directory used for most file transfers.

/usr/lib/uucp	Most of the UUCP programs are stored here, as well as the supporting database or control files. The main user programs, including **uux** and **uucp**, are found in */usr/bin*.

The */usr/lib/uucp* directory also contains configuration files for UUCP (distinguished by their capitalized names). The most important to understand are:

Systems	contains information needed to establish a link to a remote computer, including the name of the connecting device associated with the remote computer, when the computer can be reached, telephone number, login sequence, and password.
Permissions	defines the access level granted to computers when they attempt to transfer files or remotely execute commands on your computer.
Devices	contains information concerning the port name, speed, and type of the Automatic Call Units (modems), direct links, and network devices.

UUCP background programs

The **uucp** traffic is managed by three *daemons*, or supervisory programs, that run in the background, handling file transfers and command executions. (The daemons can also be executed manually as commands.)

uucico	selects the device used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers data and executes files, logs results, and (if requested) notifies the user by mail of transfer completions. When the local **uucico** daemon calls a remote computer, it "talks" to the **uucico** daemon on the remote computer during the session.
uuxqt	performs remote program execution. It searches the spool directory for execute files (*X.file*s) that were sent from a remote computer. When an *X.file* file is found, **uuxqt** opens it to get the list of data files that are required for the execution. It then checks to see if the required data files are available and accessible. **uuxqt** also verifies that it has permission to execute the requested command.
uusched	schedules the queued work in the spool directory. Before starting the **uucico** daemon, **uusched** randomizes the order in which remote computers are called.

How UUCP works

When you enter a UUCP command, the program creates a work file and usually a data file for the requested transfer. The work file contains information required for transferring the file(s). The data file is a copy of the specified source file. After these files are created in the spool directory, the **uucico** daemon is started.

The **uucico** daemon attempts to establish a connection to the remote computer. First it gathers the information required for establishing a link to the remote computer from the *Systems* file. This is how **uucico** knows what type of device to use in establishing the link. Next, **uucico** searches the *Devices* file looking for the devices that match the requirements listed in the *Systems* file. After **uucico** finds an available device, it attempts to establish the link and log in on the remote computer.

When **uucico** logs in on the remote computer, the **uucico** daemon is started on the remote computer. The two **uucico** daemons then negotiate the line protocol to be used in the file transfer(s). The local **uucico** daemon then transfers the file(s) that you are sending to the remote computer. The remote **uucico** places the file in the specified pathname(s) on the remote computer. After your local computer completes the transfer(s), the remote computer may send files that are queued for your local computer. The remote computer can be denied permission to transfer these files with an entry in the *Permissions* file. (This is also affected by directory permissions.) If this is done, the remote computer must establish a link to your local computer to perform the transfers. A remote computer can also request files.

If the remote computer or the device selected to make the connection to the remote computer is unavailable, the request remains queued in the spool directory. If set up to run by **cron** each hour, **uudemon.hour** starts the **uusched** daemon. When the **uusched** daemon starts, it searches the spool directory for the remaining work files, generates the random order in which these requests are to be processed, and then starts the transfer process (**uucico**) described in the previous paragraphs.

A sample UUCP transaction

The following steps trace the execution of a **uucp** command:

1. A user on a system called *kilgore* wishes to send a copy of the file *minutes.01.10* to a remote system called *obie.* To accomplish this, the user enters the following command:

   uucp minutes.01.10 obie\!/usr/spool/uucppublic

   Note that the exclamation point need only be escaped (preceded by a " \ ") if the **csh** is used; the Bourne shell (**sh**) and Korn Shell (**ksh**) do not require this.
2. A work file is created in the */usr/spool/uucp/obie* directory, *C.obieNxxxx,* where *xxxx* is the job number.
3. The **uusched** daemon schedules the request for execution by **uucico.**
4. When the execution time is reached, **uucico** first checks the *Systems* file and confirms that *obie* is a recognized system and that a call is permitted at this time.
5. Using the information in the *Systems* file, **uucico** next locates the modem device and tty port associated with it as stored in the *Devices* file.
6. Using the phone number in the *Systems* file and the modem type from the *Devices* file, **uucico** uses the appropriate modem commands from the *Dialers* file (or runs a dialer program from the */usr/lib/uucp* directory) to connect to the remote system.

Table 18-1 Example UUCP control files (sites: kilgore and obie)

Systems:

```
obie Any ACU 2400 14081234567 --ogin:-BREAK-ogin:
 nuucp ssword: mavra
```

Devices:

```
ACU tty1A - 2400 dialHA24
```

Permissions:

```
LOGNAME= ukilgore MACHINE= kilgore \
  READ=/usr/spool/uucppublic:/usr/kilgore \
  WRITE=/usr/spool/uucppublic:/usr/kilgore \
  REQUEST=no  SENDFILES=call \
  COMMANDS=rmail:rnews:uucp
```

7. **uucico** creates a lock file (*LCK..tty1a*) to lock the serial line, and a lock file (*LCK..obie*) to lock the called system in the directory */usr/spool/uucp.*
8. **uucico** uses the login sequence and password defined in the *Systems* file to log in to *obie,* whose own **uucico** confirms that *kilgore* is recognized before beginning the actual transaction.
9. The calling system, *kilgore,* is said to be the "guest"; the called system, *obie,* is said be the "host". The host **uucico** checks the local *Permissions* file to confirm that the guest is authorized to transfer the file.

10. The guest (*kilgore*) transmits the file in packets that are checked for errors and retransmitted if garbled. During reception, the file is stored in a temporary file (*TM.xxxx*) in the */usr/spool/uucp/kilgore* directory on the host (*obie*). When the transfer is complete, the file is moved to the proper destination, in this case */usr/spool/uucppublic/minutes.01.10.*
11. Each machine records its side of the transaction in log files. For example, *obie* would have the exchange recorded in a file called */usr/spool/uucp/.Log/uucp/kilgore.*
12. Unless the host system *obie* has requests of its own, a hangup request is sent, the connection is terminated, and the lock files are removed.

For remote command execution (via **uux**), an execute *X.file* is created in the */usr/spool/uucp* directory. The **uuxqt** daemon scans this directory for work, checks the *Permissions* file to confirm permission to execute the command, then executes it.

Configuring UUCP on your system

To configure your UUCP system, you must connect a modem and edit a series of files that contain information about, and control the actions of, the UUCP programs. The UUCP control files are in the */usr/lib/uucp* directory. You can modify these files with a standard text editor, but it is more sensible to use the **uuinstall** utility. The descriptions found in "Detailed descriptions of UUCP control files" provide details on the structure of these files so that you can create more complex configurations than the examples provided.

NOTE After configuring UUCP, if you have any problems initiating transactions, see the section on UUCP in the "Troubleshooting your system" chapter in this guide for helpful information.

Connecting a UUCP modem

To configure and install a modem, follow the instructions in the "Using modems" chapter of this guide and return to this section after your modem is up and running.

Variable rate modems

Some modems can determine the connection baud rate from the carrier sent by a remote system. These modems inform the local system of the connection baud rate before issuing the Carrier Detect (CD) signal. The Hayes 2400 dialer supplied with UUCP detects different connection baud rates and informs UUCP and **cu** when it exits with a successful connection.

The speed fields in *Devices* and *Systems* can specify a range of baud rates for a connection. If a dialer supports baud rates from 300 to 2400 baud, enter the baud rate range in the speed field of *Devices* as follows:

300-2400

If a dialer or modem does not allow variable baud rates, place a single baud rate in the speed field. If a remote system supports several different speeds, place the range of baud rates in the speed field of *Systems*. If the remote system connects at a single baud rate, place that number in *Systems*. UUCP passes the intersection of the *Systems* and *Devices* baud rate ranges to the dialer when connecting. If the dialer connects outside of the baud range, it returns a bad baud rate error. Otherwise, it returns the baud rate of the connection.

Editing the UUCP control files

This section is concerned with the configuration or control files that act as the UUCP database. A simple configuration is assumed in this section; more detailed descriptions of the UUCP files are found later in this chapter.

To configure the UUCP files, do the following:

1. Set up the *Systems* file on each machine. Use the following format:

   ```
   sitename Any ACU baud phone# -\r\d-ogin:-\K\d-ogin:-\K\d-ogin:-\K\d-ogin:
    nuucp word: password
   ```

   In the above lines:

sitename	login name for the opposite site
baud	baud rate used for dial-out
phone#	phone number of opposite site
password	password for UUCP account (nuucp) at the opposite site

   NOTE Note that the lines will exceed 80 characters in length, but should still be treated as one line each. In other words, do not press ⟨Return⟩ when the text reaches the right side of the screen. Press ⟨Return⟩ only when you have finished typing in the line.

   Sitenames should be no longer than seven characters and should contain no 8-bit characters (some sites reject both). In addition, they should not contain control characters, escape sequences, or uppercase letters.

   The baud rate should be set to the highest common baud rate between the modems that will be used. In other words, if the modem on machine A is a 2400 baud modem, and the modem on machine B is a 1200 baud modem, then these should both be set for 1200.

Here is a set of example entries for a pair of sites, kilgore and obie, with 2400 baud modems:

site: obie

```
kilgore Any ACU 2400 5551212 -\r\d-ogin:-\K\d-ogin:-\K\d-ogin:-\K\d-ogin:
 nuucp word: TrouTster
```

site: kilgore

```
obie Any ACU 2400 5551212 -\r\d-ogin:-\K\d-ogin:-\K\d-ogin:-\K\d-ogin:
 nuucp word: mAvraC
```

There are many other specifications that can be included in the *Systems* file; it is discussed in more detail in "Adding entries for remote systems to the Systems file" later in this chapter.

2. Set up the *Permissions* file on each machine. Add the following text to the bottom of the file */usr/lib/uucp/Permissions* on each machine, leaving a blank line between any text already in the file and the following new text:

```
MACHINE=site LOGNAME=login \
COMMANDS=rmail:rnews:uucp \
READ=/usr/spool/uucppublic:/usr/tmp \
WRITE=/usr/spool/uucppublic:/usr/tmp \
SENDFILES=yes REQUEST=yes
```

Note that the **LOGNAME** must be the login name that the site uses. There are many other options that can be included in the *Permissions* file; it is discussed in more detail in "Limiting access with the Permissions file" later in this chapter.

Creating login accounts for sites dialing-in

A dial-in site must provide a login entry (login account - for example, nuucp) for the sites that call it.

A UUCP login account is the same as an ordinary user account (see the "Administering user accounts" chapter in this guide), but it has a special login directory and login program instead of the normal user directory and shell.

NOTE "uucp" should not be used as the name of a UUCP user or login account; it is the name of the UUCP owner or administrator.

To create a UUCP login entry (for example, nuucp), follow these steps:

1. Choose a new user name and a user ID (identification number) for the UUCP login. The name can be any combination of letters and digits that is no more than eight characters long. The user ID must be an integer in the range 50 to 65535.

   Make sure the name and ID are unique. A UUCP login entry must not have the same name or ID as any other login entry.

2. To create the new account, invoke **sysadmsh** and make the following selection:

   Accounts ⇨ User ⇨ Create

3. Use the following information to create the account:

   Login shell: */usr/lib/uucp/uucico*
   Home directory: */usr/spool/uucppublic*

   Passwords are optional, but recommended, for UUCP logins.

UUCP anonymous login accounts

UUCP login accounts are created with a default password expiration of 14 days. To alter this, you must use the **sysadmsh**(ADM) Accounts ⇨ User ⇨ Examine:Expiration selection to redefine this limit. For more information, see the "Administering user accounts" chapter in this guide.

NOTE Remember that UUCP login accounts are used by remote systems using a login script which cannot cope with a prompt for a new password. For this reason it is sensible to set up an infinite password expiration, with the password changed manually in consultation with the remote site using that UUCP login.

If you have difficulties with UUCP accounts being locked (messages like "dead account" are displayed), you can extend the number of login attempts by selecting Accounts ⇨ User ⇨ Examine:Logins . If the account was locked due to too many unsuccessful login attempts, the "Account Locked" field displays "Too many unsuccessful login attempts". You can clear this condition either by setting the maximum unsuccessful logins to a larger number (including infinite), or by selecting the Lock status option Clear all locks.

Testing the UUCP connection

To test your UUCP connection, follow these steps:

1. If you are using a Hayes 1200 or compatible, make sure the volume switch on the modem is at an appropriate level. You must be able to hear the modem to carry out this test successfully. Refer to your modem reference manual for the location of this switch.
2. Ensure that the *Systems* file has an entry for the system you intend to call, and that the *Devices* file has a matching entry for ***ttynn***.
3. Start the **uutry** program by entering:

   /usr/lib/uucp/uutry -x6 *sitename*

4. Listen carefully to the modem. You should hear each digit as the number is dialed, then hear a high-pitched signal when the other modem connects, followed by silence.
5. The dialer automatically disconnects any call that it cannot complete. To break out of the shell created by **uutry**, press ⟨Del⟩ or ⟨BREAK⟩. This returns control to the terminal while **uucico** continues to run, sending the output to a file in */tmp* with the name of the system called.
6. If the signal is not present, make certain:
   - the modem is connected to the telephone jack
   - the jack is connected to the phone system
   - the correct phone number is in the *Systems* file
7. If you do not hear the modem dial, make certain:
   - the volume switch is up
   - the modem is connected to the correct serial line and that the cable connection is tight
   - the correct tty line is in the *Devices* file
   - the modem's power is on
   - there are no *LCK..* files in */usr/spool/uucp*.
8. The **uucico** daemon only allows one call to a given system every 10 minutes. You can wait before retrying, or remove the file associated with the site you are calling in the directory */usr/spool/uucp/.Status*. (This file maintains the status of the connection, and its presence prevents a call until the retry is scheduled).

Detailed descriptions of UUCP configuration files

This section includes detailed information on the UUCP database files. Sites with specific needs can design files as needed rather than relying on the simple examples included in this chapter.

Adding entries for remote sites to the Systems file

The *Systems* file (*/usr/lib/uucp/Systems*) contains the information needed by the **uucico** daemon to establish a communications link to a remote computer. Each entry in the file represents a computer that can be called by your computer.

NOTE If you plan to route mail traffic over UUCP, you must also configure MMDF as described in the "Setting up electronic mail" chapter of this guide.

In addition, the *Systems* file can be configured to prevent any computer that does not appear in this file from logging in on your computer. More than one entry may be present for a particular computer. The additional entries represent alternative communication paths that can be tried in sequential order.

NOTE If you are setting up your system as a dial-in only (passive) site that never initiates calls, you only need to add the names of the systems that will be calling you with the keyword "Never" as in this example:

```
guardian Never
```

Each entry in the *Systems* file has the following format (each field must be separated by a space):

sitename schedule device speed phone login-script

where:

sitename	contains the node name of the remote computer.
schedule	is a string that indicates the day-of-week and time-of-day when the remote computer can be called.
device	is the device type that should be used to establish the communications link to the remote computer.
speed	indicates the transfer speed of the device used in establishing the communications link.
phone	provides the phone number of the remote computer for automatic dialers. If you wish to create a portable *Systems* file that can be used at a number of sites where the dialing prefixes differ (for internal phone systems), refer to "Using Dialcodes to create a portable Systems file" under "Special UUCP configuration options" later in this chapter.
login-script	contains login information (also known as a "chat script").

The schedule field

The *schedule* consists of three subfields. The first, ***day***, is required. The other two, ***time*** and ***retry***, are optional. The syntax is as follows:

> ***day***[***time***][;***retry***]

The ***day*** subfield can contain the following keywords:

Su Mo Tu We Th Fr Sa	for individual days
Wk	for any weekday (Mo Tu We Th Fr)
Any	for anytime
Never	for a passive arrangement with the remote computer. If the *schedule* field is **Never**, your computer never initiates a call to the remote computer. (This field is ignored when you set up polling with **uudemon.poll2**; see "Setting up polling" for details.) The call must be initiated by the remote computer. In other words, your computer is in a passive mode with respect to the remote computer (see discussion of *Permissions* file).

The optional ***time*** subfield should be a range of times in 24-hour clock format, such as 0800-1230. If no ***time*** is specified, any time of day is assumed to be allowed for the call. A time range that spans 0000 is permitted. For example, 0800-0600 means all times are allowed other than times between 6 am and 8 am.

For example, the following permits calls on Mondays, Wednesdays, and Fridays between the hours of 9 am and noon (the ***schedule*** field is in boldface for clarity):

```
grebe MoWeFr0900-1200 ACU D1200 14087672676 ogin:
 nuucp ssword: Crested
```

You can also specify more than one set of ***day*** and ***time*** entries by separating them with commas. This is useful for more complex specifications. The following example allows calls from 5 pm to 8 am, Monday through Thursday, and calls any time on Saturday and Sunday. This example would be an effective way to call only when phone rates are low, if immediate transfer is not critical:

```
gorgon Wk1700-0800,SaSu ACU D1200 14087672676 ogin:
 nuucp ssword: DontLook
```

The optional subfield, ***retry***, is available to specify the minimum time (in minutes) before a retry following a failed attempt. The subfield separator is a semicolon (;). For example, the following is interpreted as "call any time, but wait at least 9 minutes before retrying after a failure occurs":

```
Any;9
```

By default, UUCP uses a method called exponential backoff to allow retry of failed calls. UUCP does not allow another call to go through until after the retry time has elapsed. This interval expands exponentially as the number of unsuccessful attempts increases. The ***retry*** field overrides the exponential backoff algorithm. If you set the retry field to 9, for example, UUCP allows another attempt to connect 9 minutes after each failure. The ***retry*** field cannot be set lower than 5 minutes.

UUCP does not automatically try a failed call again. You must have polling set up as described in "Setting up polling" in this chapter or manually invoke **uucico**(ADM). Any files not transferred due to a connection failure are transferred at the next successful connection to that system.

The device field

The ***device*** field selects the device type, in most cases an ACU (Automatic Call Unit). For example, the keyword used in the following field is matched against the first field of *Devices* file entries:

Systems:
```
gorgon Any ACU D1200 14087672676 ogin:
 nuucp ssword: DontLook
```

Devices:
```
ACU tty2A - 1200 /usr/lib/uucp/dialHA12
```

The speed field

This field can contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers (refer to the discussion on the *Devices* file, ***speed*** field). Some devices can be used at any speed, so the keyword **Any** can be used. However, we recommend that you specify the actual range of speeds that can be used. (If **Any** is used in both *Systems* and *Devices* entries, 1200 is assumed.) For example, this field must match the ***speed*** field in the associated *Devices* file entry:

Systems:
```
gorgon Any ACU D2400-9600 14087672676 ogin:
 nuucp ssword: DontLook
```

Devices:
```
ACU tty1A - D2400-9600 /usr/lib/uucp/dialHA9600
```

If information is not required for this field, use a hyphen (-) as a place holder for the field.

The phone field

This field provides the phone number used for the modem dialer. The phone number is made up of an optional alphabetic abbreviation and a numeric part. If an abbreviation is used, it must be one that is listed in the *Dialcodes* file. (See "Using Dialcodes to create a portable Systems file" in this chapter for details.) For example:

Systems:
```
gorgon Any ACU D1200 CA2676   ogin:
 nuucp ssword: DontLook
```

Dialcodes:
```
CA 9=408767
```

In this string, an equal sign (=) tells the ACU to wait for a secondary dial tone before dialing the remaining digits. A dash in the string (-) instructs the ACU to pause 2 seconds before dialing the next digit.

NOTE Do not use the comma (,) from the Hayes command set in a *Systems* file entry when you wish to indicate a pause. Use hyphens instead.

If your computer is connected to a LAN switch or port selector, you can access other computers that are connected to that switch. The *Systems* file entries for these computers do not have a phone number in the ***phone*** field. Instead, this field contains the token that must be passed on to the switch so it knows which computer your computer wishes to communicate with. (This is usually just the system name.) The associated *Devices* file entry should have a " \D" at the end of the entry to prevent translation using the *Dialcodes* entry.

The login-script field

The login-script opens communications between modems, and also recognizes and sends proper login and password sequences. The script is given as a series of space-separated fields and subfields of the following format:

expect send

where ***expect*** is the string that is received, and ***send*** is the string that is sent when the ***expect*** string is received.

The ***expect*** field can be made up of subfields of the following form:

expect[-***subsend-subexpect***]...

where the ***subsend*** is sent if the prior ***expect*** is not successfully read and the ***subexpect*** following the ***subsend*** is the next expected string. To make this distinction clear: the send-expect sequence sends a string if the expect string is received; the subsend-subexpect sends only if the prior expect string is not received within 10 seconds.

For example, with "login:--login:", the UUCP program expects "login:". If a "login:" is received, it goes on to the next field. If it does not get "login:", it sends nothing followed by a carriage return, then looks for "login:" again. If no characters are initially expected from the remote computer, the null string ("") should be used in the first ***expect*** field. Note that all ***send*** fields are sent followed by a carriage return unless the ***send*** string is terminated with a " \c ".

If an ***expect*** string starts with a dash, it is interpreted as a null ***expect*** string followed by a ***subsend*** string. For example, "--login:" sends a carriage return and then expects a "login:".

The ***expect*** string need not be complete; only the trailing characters must be specified, as in "ogin:". This avoids difficulties with login strings that use an uppercase letter as in "Login:" or "Password:", and also difficulties when the line is shared by dial-in and dial-out.

Creating login scripts

This section explains in greater detail how to create a login (chat) script.

Consider the following sample *Systems* file entry:

```
terps Any ACU 1200 18005211980 "" \r ogin:-BREAK-ogin:
 uucpx word: ichore
```

This is how this script would work during connection:

1. Nothing is expected initially.
2. A carriage return is sent and the script waits for the prompt "ogin:" (login:).
3. If it does not receive "ogin:", send a **BREAK** signal.
4. When "ogin:" is finally received, send the login name *uucpx*.
5. When the prompt "word:" (for Password:) is received, send the password *ichore*.

Login (chat) scripts often require some experimentation. There are cases that require one or more **BREAK** sequences before presenting a login (this is often true with variable speed modems). If you cannot obtain the necessary login sequence from the system administrator for a given site, it is a good idea to connect with the site manually. You can accomplish this using **cu** and find out what must be sent to generate a login prompt. (You can also connect with a system using a **uutry** for debugging; see the "Troubleshooting your system" chapter for details.) There are several escape characters that cause specific actions when sent during the login sequence, some of which correspond to keystrokes; these should be included in the script where necessary. See Table 18.2.

Table 18-2 Login (Chat) script escape sequences

Character	Description
\N	sends a null character (ASCII **NULL**).
\b	sends or expects a backspace character.
\c	if at the end of a string, suppresses the carriage return that is normally sent. Ignored otherwise.
\d	delays two seconds before sending or reading more characters.
\p	pauses for approximately ¼ to ½ second.
\E	starts echo checking. (After this sequence is used, whenever a character is transmitted, the system waits for the character to be received before doing anything else.)
\e	turns echo check off.
\n	sends or expects a newline character.
\r	sends or expects a carriage-return.
\s	sends or expects a space character.
\t	sends or expects a tab character.
\\\\	sends or expects a " \\ " character.
EOT	sends **EOT** (end of transmission or ⟨Ctrl⟩**d**)
BREAK	sends a **BREAK** signal.
\K	same as **BREAK.**
\ddd	collapses the octal digits (***ddd***) into a single character whose value is the ASCII character represented by that number (for example: \007).
""	expects a null string.

Setting up polling

Use **uudemon.poll2** to set up polling. To run **uudemon.poll2**, you need an entry for the daily daemon and an entry for the hourly daemon in the */usr/spool/cron/crontabs/root* file as follows:

```
0 0 * * *       uudemon.poll2 -d
0 * * * *       uudemon.poll2
```

The **-d** flag refers to the daily daemon. The above example has the daemon run at midnight. You can change the time the daemon runs by altering the second field using a 24-hour clock. The hourly daemon has no flags.

To establish the hours and days that **uudemon.poll2** runs, you create two files: */usr/lib/uucp/Poll.hour* and */usr/lib/uucp/Poll.day*. These files contain the systems to be polled and the times and days they are polled. A sample *Poll.hour* file follows:

```
hanna   12 1 3
raven   2 6 10w
```

If the hour is followed by a "w", **uudemon.poll2** calls the site only if there is work to be done.

A sample *Poll.day* file follows:

```
hanna   1 3 6
raven   1 2 3 4 5
```

The days of the week are integers where Sunday is 0.

Limiting access with the Permissions file

If other machines will be dialing into your system, the *Permissions* file (*/usr/lib/uucp/Permissions*) specifies the permissions that remote computers have with respect to login, file access, and command execution. There are options that restrict the remote computer's ability to request files and its ability to receive files queued by the local site. Other options specify the commands that a remote site can execute on the local computer.

Structuring Permissions file entries

Each entry is a logical line with physical lines terminated by a "\" to indicate continuation. Entries are made up of options delimited by spaces. Each option is a name-value pair in the following format:

name=value

Note that no spaces are allowed within an option assignment. This means that any continuations in an option assignment cannot have spaces before the "\" or at the start of the next line.

Comment lines begin with a number sign (#) and they occupy the entire line up to a newline character. Blank lines are ignored (even within multi-line entries).

There are two types of *Permissions* file entries:

LOGNAME	specifies the permissions that take effect when a remote computer calls your computer.
MACHINE	specifies permissions that take effect when your computer calls a remote computer.

In this way it is possible not only to define permissions for sites calling your system, but permissions for when your site calls other machines.

Permissions file restrictions

When using the *Permissions* file to restrict the level of access granted to remote computers:

- A machine cannot have more than one **LOGNAME** entry.
- Any site that is called whose name does not appear in a **MACHINE** entry, has the following default permissions or restrictions:
  - Only local send and receive requests are executed.
  - The remote computer can send files to your computer's */usr/spool/uucppublic* directory.
  - The commands sent by the remote computer for execution on your computer must be one of the default commands, usually **rmail**.

NOTE When a remote machine calls you, unless you have a unique login and password for that machine, you do not know if the machine is who it claims to be.

Permissions options

This section describes each option, specifies how they are used, and lists their default values.

REQUEST

specifies whether the remote computer can request to set up file transfers from your computer. When a remote computer calls your computer and requests to receive a file, this request can be granted or denied. The following string specifies that the remote computer can request to transfer files from your computer:

```
REQUEST=yes
```

The following string specifies that the remote computer cannot request to receive files from your computer:

```
REQUEST=no
```

The no value is the default value. It is used if the **REQUEST** option is not specified. The **REQUEST** option can appear in either a **LOGNAME** (remote calls you) entry or a **MACHINE** (you call remote) entry.

SENDFILES

specifies whether your computer can send the work queued for the remote computer. When a remote computer calls your computer and completes its work, it may attempt to take work your computer has queued for it.

The following string specifies that your computer can send the work that is queued for the remote computer as long as the remote computer is logged in as one of the names in the **LOGNAME** option:

```
SENDFILES=yes
```

This string is mandatory if your computer is in a passive mode with respect to the remote computer.

The following string specifies that files queued in your computer be sent only when your computer calls the remote computer:

```
SENDFILES=call
```

The call value is the default for the **SENDFILE** option. This option is only significant in **LOGNAME** entries because **MACHINE** entries apply when calls are made out to remote computers. If this option is used with a **MACHINE** entry, it is ignored.

READ and **WRITE**

specify the various parts of the filesystem that **uucico** can read from or write to. The **READ** and **WRITE** options can be used with either **MACHINE** or **LOGNAME** entries.

The default for both the **READ** and **WRITE** options is the *uucppublic* directory as shown in the following strings:

```
READ=/usr/spool/uucppublic
WRITE=/usr/spool/uucppublic
```

The following strings specify permission to access any file that can be read by UUCP.

```
READ=/  WRITE=/
```

The value of these entries is a colon-separated list of pathnames. The **READ** option is for requesting files, and the **WRITE** option for depositing files. One of the values must be the prefix of any full pathname of a file coming in or going out.

NOTE **READ** and **WRITE** options do not affect the actual permissions of a file or directory. For example, a directory with permissions of 700 only permits the owner to access it, and cannot be read or written by UUCP, no matter what access options are defined in the *Permissions* file.

To grant permission to deposit files in */usr/tmp* as well as the public directory, the following values would be used with the **WRITE** option:

```
WRITE=/usr/spool/uucppublic:/usr/tmp
```

It should be pointed out that if the **READ** and **WRITE** options are used, all pathnames must be specified because the pathnames are not added to the default list. For instance, if the */usr/news* pathname was the only one specified in a **WRITE** option, permission to deposit files in the public directory would be denied.

You should be careful with which directories you make accessible for reading and writing by remote systems. For example, you probably do not want remote computers to be able to write over your */etc/passwd* file so */etc* should not be open to writes.

NOREAD and NOWRITE

specify exceptions to the **READ** and **WRITE** options or defaults. The following strings would permit reading any file except those in the */etc* directory (and its subdirectories—remember, these are prefixes) and writing only to the default */usr/spool/uucppublic* directory:

```
READ=/
WRITE=/usr/spool/uucppublic
NOREAD=/etc
NOWRITE=/etc
```

NOWRITE works in the same manner as the **NOREAD** option. The **NOREAD** and **NOWRITE** options can be used in both **LOGNAME** and **MACHINE** entries.

CALLBACK

specifies in **LOGNAME** entries that no transaction takes place until the calling system is called back. There are two examples of when you would use **CALLBACK**. From a security standpoint, if you call back a machine you can be sure it is the machine it says it is. If you are doing long data transmissions, you can choose the machine that is billed for the longer call.

The following string specifies that your computer must call the remote computer back before any file transfers take place:

```
CALLBACK=yes
```

The default for the **CALLBACK** option is:

```
CALLBACK=no
```

The **CALLBACK** option is rarely used. If two sites have this option set for each other, a conversation never gets started.

COMMANDS

specifies the commands in **MACHINE** entries that a remote computer can execute on your computer. This affects the security of your system; use it with extreme care.

The **uux** program generates remote execution requests and queues them to be transferred to the remote computer. Files and a command are sent to the target computer for remote execution. Note that **COMMANDS** is not used in a **LOGNAME** entry; **COMMANDS** in **MACHINE** entries define command permissions whether you call the remote system or it calls you.

The default command that a remote computer can execute on your computer is:

```
COMMANDS=rmail
```

If a command string is used in a **MACHINE** entry, the default commands are overridden. For instance, the following entry overrides the **COMMAND** default so that the computers *owl*, *raven*, *hawk*, and *dove* can now execute **rmail, rnews,** and **lp** on your computer:

```
MACHINE=owl:raven:hawk:dove \
COMMANDS=rmail:rnews:lp
```

Full pathnames of commands can also be used. For example, the following command specifies that command **rmail** uses the default path:

```
COMMANDS=rmail:/usr/lbin/rnews:/usr/local/lp
```

The default paths for your computer are */bin*, */usr/bin*, and */usr/lbin*. When the remote machine specifies **rnews** or */usr/lbin/rnews* for the command to be executed, */usr/lbin/rnews* is executed regardless of the default path. Likewise, */usr/local/lp* is the **lp** command that is executed.

Including the **ALL** value in the list means that any command from the remote computer specified in the entry is executed. If you use this value, you give the remote computer full access to your computer. So, be careful; this allows far more access than normal users have.

The following string illustrates two points:

```
COMMANDS=/usr/local/bin/lc:ALL:/usr/local/lp
```

1. The **ALL** value can appear anywhere in the string; and the pathnames specified for **lc** and **lp** are used (instead of the default) if the requested command does not contain the full pathnames for **lc** or **lp**.

2. The **VALIDATE** option should be used with the **COMMANDS** option whenever potentially dangerous commands like **cat** and **uucp** are specified with the **COMMANDS** option. Any command that reads or writes files is potentially dangerous to local security when executed by the UUCP remote execution daemon (**uuxqt**).

VALIDATE

is used in conjunction with the **COMMANDS** option in **LOGNAME** entries when specifying commands that are potentially dangerous to your computer's security. It provides a certain degree of verification of the caller's identity. The use of the **VALIDATE** option requires that privileged computers have a unique login or password for UUCP transactions. An important aspect of this validation is that the login or password associated with this entry be protected. If an outsider gets that information, that particular **VALIDATE** option can no longer be considered secure. (**VALIDATE** is merely an added level of security to the **COMMANDS** option, though it is a more secure way to open command access than **ALL**.)

Careful consideration should be given to providing a remote computer with a privileged login and password for UUCP transactions. Giving a remote computer a special login and password with file access and remote execution capability is like giving anyone on that computer a normal login and password on your computer. Therefore, if you cannot trust someone on the remote computer, do not provide that computer with a privileged login and password.

The following **LOGNAME** entry specifies that if one of the remote computers that claims to be *eagle, owl,* or *hawk* logs in on your computer, it must have used the login *uucpfriend.*

```
LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

As can be seen, if an outsider gets the *uucpfriend* login or password, masquerading is trivial.

VALIDATE increases security by linking the **MACHINE** entry (and **COMMANDS** option) with a **LOGNAME** entry associated with a privileged login. This link is needed because the execution daemon is not running while the remote machine is logged in. In fact, it is an asynchronous process with no knowledge of what machine sent the execution request. Therefore, the real question is how does your system know where the execution files came from?

Each remote computer has its own *spool* directory on your computer. These spool directories have write permission given only to UUCP programs. The execution files from the remote computer are put in its spool directory after being transferred to your computer. When the **uuxqt** daemon runs, it can use the spool directory name to find the **MACHINE** entry in the *Permissions* file and get the **COMMANDS** list. If the computer name does not appear in the *Permissions* file, the default list is used.

The following example shows the relationship between the **MACHINE** and **LOGNAME** entries:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
COMMANDS=rmail:/usr/local/bin/lc \
READ=/  WRITE=/

LOGNAME=uucpz VALIDATE=eagle:owl:hawk \
REQUEST=yes SENDFILES=yes \
READ=/  WRITE=/
```

The **COMMANDS** option line shows that remote mail and */usr/local/bin/lc* can be executed by remote users.

In the **MACHINE** entry, you must make the assumption that when you want to call one of the computers listed, you are really calling *eagle, owl,* or *hawk.* Any files put into one of the *eagle, owl,* or *hawk* spool directories is put there by one of those computers. If a remote computer logs in and says that it is one of these three computers, its execution files are also put in the privileged spool directory. You should validate that the computer has the privileged login *uucpz.*

Entries for OTHER Systems

You may want to specify different option values for machines or logins that are not mentioned in specific **MACHINE** or **LOGNAME** entries. This may occur when there are many computers calling in that have the same set of permissions. The special name **OTHER** for the computer name can be used in a **MACHINE** or **LOGNAME** entry as follows:

```
MACHINE=OTHER \
COMMANDS=rmail:/usr/local/bin/lc

LOGNAME=OTHER \
REQUEST=yes SENDFILES=yes \
READ=/usr/spool/uucppublic \
WRITE=/usr/spool/uucppublic
```

All options that can be set for specific machines or logins can be used with the **OTHER** value, although the use of the **VALIDATE** option makes little sense.

Combining **MACHINE** and **LOGNAME** Entries

It is possible to combine **MACHINE** and **LOGNAME** entries into a single entry where the common options are the same. For example, the following two entries share the same **REQUEST, READ,** and **WRITE** options:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
  READ=/  WRITE=/

LOGNAME=uucpz REQUEST=yes SENDFILES=yes \
  READ=/  WRITE=/
```

These two entries can be merged as follows:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
LOGNAME=uucpz SENDFILES=yes \
  READ=/  WRITE=/
```

Adding dial-out entries to the Devices file

The *Devices* file (*/usr/lib/uucp/Devices*) contains information for all the devices that can be used to establish a link to a remote computer. Devices are Automatic Call Units, direct links, or network connections. This file works closely with the *Dialers, Systems,* and *Dialcodes* files. Before you make changes in any of these files, you should be familiar with them all. A change to an entry in one file may require a change to a related entry in another file.

Each entry in the *Devices* file has the following format:

type ttyline dialerline speed dialer-token

where:

type	contains one of two keywords (**direct** or **ACU**), the name of a Local Area Network switch, or a system name.
ttyline	contains the device name of the port associated with the *Devices* entry. For example, if the automatic dial modem for a particular entry was attached to the */dev/tty1A* line, the name entered in this field would be *tty1A*.
dialerline	is useful only for 801 type dialers, which do not contain a modem and must use an additional line. Unless you have an 801 dialer, simply enter a hyphen (-) as a placeholder.
speed	is the speed or speed range of the device. Can also contain an indicator for distinguishing different dialer classes.
dialer-token	contains pairs of dialers and tokens, each representing a dialer and an argument to be passed to it. The ***dialer*** portion can be the name of an automatic dial modem, or **Direct** for a direct link device.

The type field

This field contains one of two keywords (**Direct** or **ACU**), the name of a Local Area Network switch, or a system name:

Direct	indicates a direct link to another computer or a switch for **cu** connections.
ACU	indicates that the link to a remote computer is made through an Automatic Call Unit. This modem can be connected either directly to your computer or indirectly through a Local Area Network (LAN) switch.
LANswitch	can be replaced by the name of a LAN switch. **micom** and **develcon** are supplied with caller scripts in the *Dialers* file.
sysname	indicates a direct link to a particular computer. (***sysname*** is replaced by the name of the computer.) This means that the line associated with this *Devices* entry is for a particular computer in the *Systems* file.

For example, the keyword **gorgon** used in the ***type*** field of the *Devices* file is matched against the third field of the *Systems* file entry:

```
Devices: gorgon tty1a - 1200 direct

Systems: gorgon Any gorgon 1200 - ogin: nuucp ssword: DontLook
```

The speed field

In most cases, this is simply the speed of the device, if the keyword **ACU** or **Direct** is used in the ***type*** field. However, ***speed*** can contain a letter and a speed (for example, C1200, D1200) to differentiate between classes of dialers (Centrex or Dimension PBX). This is necessary because many larger offices may have more than one type of telephone network: one network may be dedicated to serving only internal office communications, while another handles the external communications. It is necessary to distinguish which lines are used for internal communications and which are used for external communications. The keyword used in the ***speed*** field of the *Devices* file is matched against the fourth field of the *Systems* file entries, for example:

```
Devices: ACU tty1A - D1200 hayes1200

Systems: gorgon Any ACU D1200 3251 ogin: nuucp ssword: DontLook
```

Some devices can be used at any speed, so the keyword **Any** can be used in the ***speed*** field. If **Any** is used, the line matches any speed requested in a *Systems* file entry. If this field is **Any** and the *Systems* file ***speed*** field is **Any**, the speed defaults to 1200 bps. If a device can be used at a range of speeds, then the speed field can specify this range (for example, 1200-9600 or D1200-9600). This is preferable to the use of **Any**.

The dialer-token field

This field has the following format:

> ***dialer*** [***token dialer token*** ...]

For a direct line, this field contains simply the word **direct**, and no token is required.

For a simple connection to a dialer, this field contains the name of the dialer, and the token is omitted; by default it is taken from the phone number field of the *Systems* file entry.

For a dialer or a network dataswitch, this field contains the name of an entry found in the *Dialers* file (**develcon** and **micom** are examples of network data switches). Other dialer types are supported by binaries instead of *Dialers* entries. (Support for 801-type dialers is provided through use of separate lines for data and the dialer. See the *Devices* file for details.) UUCP recognizes a dialer as a binary if the name begins with a " / " or there is an executable file by that name in */usr/lib/uucp.*

Table 18.3 lists dialer types that are available as *Dialers* entries.

Table 18-3 Dialers file entries

Dialer type	Modem or Data switch
Direct	direct line; no dialer
Penril	Penril modem
Hayes	Hayes modem (or compatible)
Ventel	Ventel 212+ modem
Vadic	Racal Vadic 3451 modem
Vadic9600	Vadic 9600VP
LANswitch	network switch described in type field
Hayes1200	Hayes Smartmodem 1200
Hayes2400	Hayes Smartmodem 2400
Develcon	Develcon network dataswitch
Micom	Micom network dataswitch
Rixon	Rixon Intelligent Modem
ATT4000	AT&T Programmable 300/1200 Modem Model 4000
ATT2212c	AT&T DATAPHONE II 2212C Modem
ATT2224	AT&T DATAPHONE II 2224 Modem
NLS	Network Listener Service

NOTE For best results, dialer programs are preferred over *Dialers* entries. The following entry is an example of an entry using a dialer binary:

```
    ACU  ttynn  -  300-2400  /usr/lib/uucp/dialHA24
```

The TLI and TLIS dialer types are currently not available.

Table 18.4 lists the binary types available in *usr/lib/uucp.*

Table 18-4 Dialer binaries

Binary file	Modem
dialHA12	Hayes Smartmodem 1200 or compatible
dialHA24	Hayes Smartmodem 2400 or compatible
dialHA96V	Hayes Smartmodem 9600 or compatible
dialMUL	Multitech Multimodem 224 EH
dialVA3450	Racal Vadic 3451 modem
dialTBIT	Telebit Trailblazer Modem
dialT1500	Trailblazer TB1500

The source is provided for these dialer binaries; you can adapt and compile your own dialers if desired. See the section "Dialing out from your computer" in the "Using modems" chapter of this guide for details.

Structuring dialer-token entries

The ***dialer-token*** can be structured four different ways, depending on the device associated with the entry:

- Simple modem connection

  If an automatic dialing modem is connected directly to a port on your computer, the ***dialer-token*** field of the associated *Devices* file entry only has one pair. This pair would normally be the name of the modem. This name matches the particular *Devices* file entry with an entry in the *Dialers* file. Therefore, the ***dialer*** field must match the first field of the following *Dialers* file entry:

  Devices: `ACU tty1A - 1200` **`ventel`**

  Dialers: **`ventel`** `=&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!`

  Notice that only the ***dialer*** portion (**ventel**) is present in the ***dialer-token*** field of the *Devices* file entry. This means that the ***token*** to be passed on to the dialer (in this case the phone number) is taken from the ***Phone*** field of a *Systems* file entry. (" \T " is implied; see the last item, "Modems used with a local network switch".) Backslash sequences are described later.

- Direct links

  If a direct-link is established to a particular computer, the ***dialer-token*** field of the associated entry contains the keyword **direct**. This is true for both types of direct link entries, **direct** and ***sysname*** (refer to discussion on the ***type*** field).

- Local network switches

  If a computer that you wish to communicate with is on the same local network switch as your computer, your computer must first access the switch and the switch can then make the connection to the other computer. In this type of entry, there is only one pair. The ***dialer*** portion matches a *Dialers* file entry, as shown in the following example:

  Devices: `develcon tty13 - 1200 develcon \D`

  Dialers: `develcon "" "" \pr\ps\c est:\007 \E\D\e \007`

  Systems:
  ```
  obie Any ACU 1200 obie --ogin:-BREAK-ogin:
     nuucp ssword: mavra
  ```

  As shown, the ***token*** portion is "\D", which indicates that it is retrieved from the *Systems* file without translation. The *Systems* file entry for this particular computer contains the token in the ***phone*** field; this is normally reserved for the phone number of the computer (refer to *Systems* file, ***phone*** field). The "\D" ensures that the contents of the ***phone*** field is not interpreted as a valid entry in the *Dialcodes* file.

- Modems used with a local network switch

  If an automatic dialing modem is connected to a switch, your computer must first access the switch and the switch makes the connection to the automatic dialing modem. This type of entry requires two ***dialer-token-pairs***. The following ***dialer*** portion of each pair (fifth and seventh fields of entry) are used to match entries in the *Dialers* file:

  Devices: `ACU tty14 - 1200` **`develcon vent ventel`**

  Dialers:
  ```
  develcon "" "" \pr\ps\c est:\007 \E\D\e \007
    ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
  ```

  In the first pair, **develcon** is the switch and **vent** is the token that is passed to the **develcon** switch to tell it which device to connect to your computer. This token would be unique for each LAN switch because each switch can be set up differently. Once the modem is connected, the second pair is accessed, where **ventel** is the dialer and the token is retrieved from the *Systems* file.

The following are two escape characters that can appear in the *dialer-token* field:

\T indicates that the ***Phone*** field should be translated at this stage, using the *Dialcodes* file. This escape character is normally placed in the *Dialers* file for each caller script associated with an automatic dial modem (penril, ventel, and so on). The translation does not take place until the caller script is accessed.

\D indicates that the ***Phone*** field should not be translated using the *Dialcodes* file. If no escape character is specified at the end of a *Devices* entry, "\D" is assumed by default when a *Dialers* script is to be used (which can itself contain a "\T" to translate the number). "\T" is assumed if a built-in or dialer binary is to be used (because there is then no later opportunity to translate the number).

Special UUCP configuration options

This section contains several options that are used for special circumstances and can be ignored in most cases.

Adding dialers to the Dialers file

The *Dialers* file (*/usr/lib/uucp/Dialers*) specifies the initial conversation that must take place on a line before it can be made available for transferring data. New entries can be added to this file if your modem does not appear in the file. See the **dialers**(F) manual page for more information.

Using Dialcodes to create a portable Systems file

The *Dialcodes* file (*/usr/lib/uucp/Dialcodes*) contains the dial-code abbreviations that can be used in the ***Phone*** field of the *Systems* file. This feature is intended primarily for those who wish to create a standard *Systems* file for distribution among several sites that have different phone systems and area codes. As such, the *Dialcodes* file is probably not necessary for most sites. See the **dialcodes**(F) manual page for more information.

Creating alternate control files with Sysfiles

The */usr/lib/uucp/Sysfiles* file lets you assign different files to be used by **uucp** and **cu** as *Systems*, *Devices*, and *Dialers* files. See the **sysfiles**(F) manual page for more information.

Changing uucico packet parameters

An added feature is the ability to change two specialized parameters contained in the **uucico** program without having to recompile the source. These parameters are:

windows	specifies the size of window that the sliding-window protocol should use.
pktime	is the number of seconds **uucico** should wait before giving up and re-transmitting the packet being sent.

See the **uucico**(ADM) manual page for more information.

Preventing unknown sites from logging in

The script **remote.unknown** is executed when a site whose name is not recognized dials in to your system. It logs the conversation attempt and fails to make a connection. If you wish to allow such "unknown" systems to log in to your system, you can change the permissions of this file so it cannot execute and your system accepts any communication requests. To do so, enter the following commands while logged in as *root*:

```
cd /usr/lib/uucp
chmod 000 remote.unknown
```

Connecting two local systems using a direct wire

This section describes how to install a direct wire between two computers. If you are using UUCP to connect remote machines, you can skip this section. To connect two computers with a direct wire, you need to do the following:

- Choose a serial port on each machine.
- Connect a serial wire (RS-232) between the two machines, using the chosen serial ports.
- Decide which machine is the dial-in site and which is the dial-out site. The dial-out site calls and logs in to the dial-in site.

Choosing a serial port

On each machine, you must choose the RS-232 serial port (*/dev/ttynn*) you want to use. If there are no ports available, you must install a new serial line or make one available by removing any device connected to it. If you remove a terminal, make sure no one is logged in.

Find the name of the device special file associated with the line. (Refer to the "UNIX directories and special device files" appendix in this guide.) The device name should have the form:

/dev/tty*nn*

where *nn* is the number of the corresponding line. For example, */dev/tty1a* usually corresponds to COM1. You need the name of the actual line for later steps.

The serial port should be owned by *uucp*. To make sure the line is owned by *uucp*, enter this command:

```
chown uucp /dev/ttynn
```

where *nn* is the number of the corresponding line.

Connecting a serial cable

You connect two computers together using an RS-232 cable. The actual pin configurations sometimes vary between machines.

Typically, the cable should connect pins 2, 3, and 7 on one computer to the same pins on the second computer. Sometimes the cable must be *nulled*, which means that pin 2 on one machine is connected to pin 3 on the other, and vice versa. Because the connections can vary, check the hardware manuals for each computer to determine the proper pin connections.

Testing a connection

For this section, *tty2a* is used as the example serial port for both machines.

To test the wire connection between two machines:

1. Disable the serial lines on each machine. On each computer, enter the command:

   ```
   disable /dev/tty2a
   ```

   Be sure to disable the modem control line as well:

   ```
   disable /dev/tty2A
   ```

2. Attach one end of the serial wire to one of the machines. Attach the other end to the standard data port of a terminal.

3. Enter this command at your computer:

   ```
   (stty 9600; date ) < /dev/tty2a > /dev/tty2a
   ```

   tty2a is our example serial line, and the date command provides sample output.

   You should see the output of the **date** command appear on the terminal screen. Repeat this procedure on the other machine.

   If this does not work, check the following:

   - The wire is plugged in properly at each end.
   - The continuity of the wire.
   - The terminal is configured correctly (baud rate, parity, and so on).
   - The serial line is disabled.
   - You are using the correct pin numbers.

NOTE An unterminated serial cable can cause serious system problems. Do not leave serial cables dangling.

Complete UUCP examples

This section includes two complete working examples of a UUCP system and the database files.

Example 1: system gomer

The following system (*gomer*) has:

- 1200 baud modem on *tty2B*
- direct connection to system (*poker* - configured for dial-in) on *tty2d* for call-out only
- three valid UUCP logins:

nuucp	The public login for email. No password required.
ubarn	The on-site login for system (*poker*).
upay4	The private login for email and file transfers.

All lines beginning with "#" are comments and are not required. Most examples are partial listings and may contain other entries. The modem answers at 1200 baud first and is set up for both call in and out.

NOTE The lines from */etc/passwd* are included here for information only. You must use the **sysadmsh**(ADM) Accounts ⇨ User ⇨ Create or Accounts ⇨ User ⇨ Modify selections to create or alter UUCP login accounts.

/etc/passwd

```
uucp:*:5:5:Uucp admin:/usr/lib/uucp:
nuucp::201:5:public:/usr/spool/uucplogins/nuucp:/usr/lib/uucp/uucico
upay4:*:202:5:private:/usr/spool/uucppublic:/usr/lib/uucp/uucico
ubarn:*:203:5:poker:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

/etc/group

```
uucp:x:5:uucp,nuucp,ubarn,upay4
```

/etc/systemid

```
gomer
```

/etc/inittab

```
Se2B:23:respawn:/etc/getty -t60 tty2B 1200
Se2b:23:off:/etc/getty tty2b m
Se2D:23:off:/etc/getty -t60 tty2D 1200
Se2d:23:off:/etc/getty tty2d m
```

/usr/lib/uucp/Devices

```
# 300-1200 baud hayes 1200 baud modem.
# The Direct tty4b entry is for programming the modem.
ACU     tty2B  -  300-1200  dialHA12
Direct  tty2b  -  300-1200  direct
poker   tty2d  -  9600      direct
```

/usr/lib/uucp/Permissions

```
# Public uucp login for mail only.
# Can send mail, transfer files to/from uucppublic, and get
# a directory (ls) listing.
LOGNAME=nuucp     MACHINE=OTHER \
        COMMANDS=rmail:ls:uucp \
        READ=/usr/spool/uucppublic:/usr/tmp \
        WRITE=/usr/spool/uucppublic:/usr/tmp \
        SENDFILES=yes   REQUEST=yes
# Private uucp login for mail and file transfer.
# Only dingbat, ogre, grinch, ... can use this login.
LOGNAME=upay4     VALIDATE=dingbat:ogre:grinch:blitzen \
        COMMANDS=rmail:ls:uucp:who:uux \
        READ=/  WRITE=/ \
        NOREAD=/etc \
        SENDFILES=yes   REQUEST=yes
# Local trusted connection to gomer
# Only poker can use this login.
LOGNAME=ubarn     VALIDATE=poker \
        COMMANDS=ALL \
        READ=/  WRITE=/ \
        SENDFILES=yes   REQUEST=yes
```

/usr/lib/uucp/Systems

```
# local calls
dingbat Any ACU 1200 4444444 ogin:-BREAK-ogin:-BREAK-ogin:
  uubig word: wetrot
# long distance (evening calls only)
grinch Any1800-0700 ACU 1200 18888888 "" \r ogin:-BREAK-ogin:
  -BREAK-ogin:nuucp
uunet Any1800-0700 ACU 1200 17031111111 ogin:-BREAK-ogin:
  -BREAK-ogin:xytpq sword: grm5q
# systems that call in as nuucp (for mail) but NOT call out.
daboss Never
sales Never
guru2 Nevea
poker Never
ogre Never
blitzen Never
```

Example 2: system dingbat

The following system (*dingbat*) has:

- 2400 baud modem on *tty1A*
- two valid UUCP logins:

nuucp	The public login for email. No password required.
uubig	The private login for email and file transfers.

All lines beginning with "#" are comments and are not required. Most examples are partial listings and may contain other entries. The modem answers at 2400 baud first and is set up for both call in and out.

/etc/passwd

```
uucp:*:5:5:Uucp admin:/usr/lib/uucp:
nuucp:*:201:5:public:/usr/spool/uucplogins/nuucp:/usr/lib/uucp/uucico
uubig:*:202:5:private:/usr/spool/uucppublic:/usr/lib/uucp/uucico
```

/etc/group

```
uucp:x:5:uucp,nuucp,uubig
```

/etc/systemid

```
dingbat
```

/etc/inittab

```
Se1A:2:respawn:/etc/getty -t60 tty1A 2400
Se1a:2:off:/etc/getty tty1a m
```

/usr/lib/uucp/Devices

```
# 300-2400 baud hayes 2400 baud modem.
# The Direct entry is for programming the modem.
ACU  tty1A  -  300-2400  dialHA24
Direct  tty1a  -  300-2400  direct
```

/usr/lib/uucp/Permissions

```
# Public uucp login for mail only.
# Can send mail, transfer files to/from uucppublic, and get
# a directory (ls) listing.
LOGNAME=nuucp     MACHINE=OTHER \
        COMMANDS=rmail:ls:uucp \
        READ=/usr/spool/uucppublic:/usr/tmp \
        WRITE=/usr/spool/uucppublic:/usr/tmp \
        SENDFILES=yes   REQUEST=yes
# Private uucp login for mail and file transfer.
# Only ogre, grinch, ... can use this login.
LOGNAME=uubig     VALIDATE=ogre:grinch:gomer:blitzen \
        COMMANDS=rmail:ls:uucp:who:uux \
        READ=/  WRITE=/ \
        NOREAD=/etc \
        SENDFILES=yes   REQUEST=yes
```

/usr/lib/uucp/Systems

```
# local calls
gomer Any ACU 2400 3333333 ogin:-BREAK-ogin:-BREAK-ogin:
  upay4 word: dryrot
# long distance (evening calls only)
grinch Any1800-0700 ACU 2400 18888888 "" \r ogin:
  -BREAK-ogin:-BREAK-ogin: nuucp
# systems that call in as nuucp (for mail) but NOT call out.
daboss Never
damgr Never
guru2 Never
ogre Never
blitzen Never
```

Sample commands

Send mail to another system and have it send the mail back. In the Bourne/Korn shell:

mail othersystem!mysystem!mylogin

In the C-shell:

mail othersystem\!mysystem\!mylogin

Print your system's full mail address. In the C-shell:

echo "\`uuname -l\`\!\`logname\`"

Display the systems you can call:

uuname

Force a call to another system and save the debug output in background:

/usr/lib/uucp/uutry -r -x7 *system*

system is the name of the system you want to debug. The output is saved to a file with the same name as the system in *tmp* directory.

Administering your UUCP system

This section discusses the various shell scripts that supervise and maintain UUCP. Consult the section on "Administration and maintenance commands" for details on all commands available to the system administrator. Included is an extended description of the */usr/spool/uucp* work directory and a special subsection on troubleshooting.

UUCP maintenance shell scripts

There are several aspects of system operation that are governed by shell scripts running as daemons:

- checking the UUCP directory for work (**uudemon.hour**)
- polling of sites that are passive or do not originate calls (**uudemon.poll2**)
- sending of status information to the UUCP administrator (**uudemon.admin**)
- cleaning of the UUCP spool directory (**uudemon.clean**)

These scripts must be set up by the system administrator. See **uudemon**(ADM) for complete instructions.

Generating log reports on usage: uulog

The **uulog** program displays log information on UUCP usage according to remote machine. All usage of the programs **uucp, uuto,** and **uux** are logged in special log files, one per machine.

uulog options

The **uulog** command has the following options:

-f*system*	displays the last entry or entries of the *system* file transfer log.
-s*system*	displays the *system* file transfer information.
-x*system*	displays the **uuxqt** log file for the given system.
-*number*	specifies the *number* of lines displayed by the **-f** option.

For example, to print the last 10 lines of *chicago*'s file-transfer log, you would enter:

```
uulog -fchicago -10
```

Special uulog files

During execution of the **uulog** program, the files from the following directories are examined:

*/usr/spool/uucp/.Log/uucico/**
: directory used for log files by the **uucico** program

*/usr/spool/uucp/.Log/uuxqt/**
: directory used for log files by the **uuxqt** program

The UUCP spool directory

The following is a comprehensive discussion of all files and subdirectories of the UUCP spool directory. These files are created in spool directories to lock devices, hold temporary data, or keep information about remote transfers or executions.

TM. (temporary data file)

These data files are created by UUCP processes under the spool directory (that is, */usr/spool/uucp/**system***) when a file is received from another computer. The ***system*** directory has the same name as the remote computer that is sending the file. The names of the temporary data files have the format:

*TM.**pid.ddd***

where ***pid*** is a process-ID and ***ddd*** is a sequential three-digit number starting at 0.

When the entire file is received, the *TM.**pid.ddd*** file is moved to the pathname specified in the *C.**sysnxxxx*** file (discussed below) that caused the transmission. If processing is abnormally terminated, the *TM.**pid.ddd*** file may remain in the ***system*** directory. These files should be automatically removed by **uuclean**.

LCK.(lock file)

Lock files are created in the */usr/spool/uucp* directory for each device in use. Lock files prevent duplicate conversations and multiple attempts to use the same calling device. The names of lock files have the format:

*LCK..**str***

where ***str*** is either a device or computer name. These files may remain in the spool directory if the communications link is unexpectedly dropped (usually on computer crashes). The lock files are ignored (removed) after the parent process is no longer active. The lock file contains the process ID of the process that created the lock. The lock file is always named by converting the last letter to lowercase (meaning non-modem control) to avoid possible conflicts if the same line is specified both as modem-control and non-modem-control. For example, the lock on */dev/tty1A* is named *LCK..tty1a.*

C.(work file)

Work files are created in a spool directory on the local computer when work (file transfer or remote command execution) is queued for a remote computer. The names of work files have the format:

C.sysnxxxx

where *sys* is the name of the remote computer, ***n*** is the ASCII character representing the grade (priority) of the work, and ***xxxx*** is the four-digit job sequence number assigned by UUCP. Work files contain the following information:

- full pathname of the file to be sent or requested
- full pathname of the destination or *~**user/filename**; "~"* is shorthand for */usr/spool/uucppublic* and must be included if the full pathname is not used.
- user login name
- list of options
- name of associated data file in the spool directory. If the **uucp -c** or **uuto -p** option was specified, a dummy name (*D.0*) is used
- mode bits of the source file
- remote user's login name to be notified upon completion of the transfer

D.(data file)

Data files are created in the spool directory on both the local and remote computers when it is specified in the command line to copy the source file to the spool directory. The names of data files have the following format:

D.systmxxxxyyy

where ***systm*** is the first five characters in the name of the remote computer, *xxxx* is a four-digit job sequence number assigned by **uucp**. The four-digit job sequence number may be followed by a subsequence number, *yyy*, that is used when there are several *D.* files created for a work (C.) file.

X.(execute file)

Execute files are created in the spool directory on the remote computer prior to remote command executions. The names of execute files have the following format:

X.sysnxxxx

where *sys* is the name of the remote computer, *n* is the character representing the grade (priority) of the work, and *xxxx* is a four-digit sequence number assigned by UUCP. Execute files contain the following information:

- Requester's login and computer name
- Name of file(s) required for execution
- Input to be used as the standard input to the command string
- Computer and file name to receive standard output from the command execution
- Command string
- Option lines for return status requests

UUCP error messages

This section lists the error messages associated with UUCP. There are two types of error messages. **ASSERT** errors are recorded in the */usr/spool/uucp/.Admin/errors* file. **STATUS** errors are recorded in individual machine files found in the */usr/spool/uucp/.Status* directory.

ASSERT error messages

When a process is aborted, ASSERT error messages are recorded in */usr/spool/uucp/.Admin/errors.* These messages include the filename, SCCS ID, line number, and the text listed in these messages. In most cases, these errors are the result of filesystem problems. Use **errno** (when present) to investigate the problem. If **errno** is present in a message, it is shown as " () " in this list.

Table 18-5 ASSERT error messages

Error Message	Description or Action
CAN'T OPEN	An **open**( ) or **fopen**( ) failed. Check for the presence of the file and incorrect permissions.
CAN'T WRITE	A **write**( ), **fwrite**( ), **fprint**( ), and so on failed. Check for the presence of the file and incorrect permissions.
CAN'T READ	A **read**( ), **fgets**( ), and so on failed. Check for the presence of the file and incorrect permissions.
CAN'T CREATE	A **create**( ) call failed. Check permissions.
CAN'T ALLOCATE	A dynamic allocation failed.
CAN'T LOCK	An attempt to make a LCK (lock) file failed. In some cases, this is a fatal error.
CAN'T STAT	A **stat**( ) call failed. Check for the presence of the file and incorrect permissions.
CAN'T CHMOD	A **chmod**( ) call failed. Check for the presence of the file and incorrect permissions.
CAN'T LINK	A **link**( ) call failed. Check for the presence of the file and incorrect permissions.

(Continued on next page)

Table 18-5 ASSERT error messages
(Continued)

Error Message	Description or Action
CAN'T CHDIR	A **chdir**() call failed. Check for the presence of the file and incorrect permissions.
CAN'T UNLINK	An **unlink**() call failed.
WRONG ROLE	This is an internal logic problem.
CAN'T MOVE TO CORRUPTDIR	An attempt to move some bad *C.* or *X.* files to the */usr/spool/uucp/.Corrupt* directory failed. The directory is probably missing or has wrong modes or owner.
CAN'T CLOSE	A **close**( ) or **fclose**( ) call failed.
FILE EXISTS	The creation of a *C.* or *D.* file is attempted, but the file exists. This occurs when there is a problem with the sequence file access. Usually indicates a software error.
No uucp server	A TCP/IP call is attempted, but there is no server for UUCP.
BAD UID	The uid cannot be found in the */etc/passwd* file. The filesystem is in trouble, or the */etc/passwd* file is inconsistent.
ULIMIT TOO SMALL	The **ulimit** for the current user process is too small. File transfers may fail, so transfer is not attempted.
BAD LINE	There is a bad line in the *Devices* file; there are not enough arguments on one or more lines.
FSTAT FAILED IN EWRDATA	There is something wrong with the Ethernet media.
SYSLST OVERFLOW	An internal table in *gename.c* overflowed. A big or strange request was attempted.
TOO MANY SAVED C FILES	Same as previous message.

(Continued on next page)

Table 18-5 ASSERT error messages
(Continued)

Error Message	Description or Action
RETURN FROM fixline ioctl	An **ioctl**, which should never fail, failed. There is a system driver problem.
BAD SPEED	A bad line speed appears in the *Devices* or *Systems* file ("Class" field).
PERMISSIONS file: BAD OPTION	There is a bad line or option in the *Permissions* file.
PKCGET READ	The remote machine probably hung up. No action need be taken.
PKXSTART	The remote machine aborted in a non-recoverable way. This can generally be ignored.
SYSTAT OPEN FAIL	There is a problem with the modes of */usr/lib/uucp/.Status*, or there is a file with bad modes in the directory.
TOO MANY LOCKS	There is an internal problem!
XMV ERROR	There is a problem with some file or directory. It is probably the spool directory, because the modes of the destinations were suppose to be checked before this process was attempted.
CAN'T FORK	An attempt to fork and exec failed. The current job should not be lost, but is attempted later (**uuxqt**). No action need be taken.

UUCP STATUS error messages

Status error messages are messages that are stored in the */usr/spool/uucp/.Status* directory. This directory contains a separate file for each remote machine that your system attempts to communicate with. These individual machine files contain status information on the attempted communication, and whether it was successful or not. What follows is a list of the most common error messages that can appear in these files.

Table 18-6 STATUS error messages

Error Message	Description or Action
OK	Self explanatory.
NO DEVICES AVAILABLE	There is currently no device available for the call. Check to see that there is a valid device in the *Devices* file for the particular system. Check the *Systems* file for the device to be used to call the system.
WRONG TIME TO CALL	A call was placed to the system at a time other than what is specified in the *Systems* file.
TALKING	Self explanatory.
LOGIN FAILED	The login for the given machine failed. It could be a wrong login or password, wrong number, a very slow machine, or failure in getting through the *dialer-token* script.
CONVERSATION FAILED	The conversation failed after successful startup. This usually means that one side went down, the program aborted, or the line (link) was dropped.
DIAL FAILED	The remote machine never answered. It could be a bad dialer or the wrong phone number.
BAD LOGIN/MACHINE COMBINATION	The machine called us with a login or machine name that does not agree with the *Permissions* file. This could be an attempt to masquerade!
DEVICE LOCKED	The calling device to be used is currently locked and in use by another process.
ASSERT ERROR	An **ASSERT** error occurred. Check the */usr/spool/uucp/.Admin/errors* file for the error message and refer to the section "ASSERT error messages".
SYSTEM NOT IN Systems	The system is not in the *Systems* file.
CAN'T ACCESS DEVICE	The device tried does not exist or the modes are wrong. Check the appropriate entries in the *Systems* and *Devices* files.

(Continued on next page)

Table 18-6 STATUS error messages
(Continued)

Error Message	Description or Action
DEVICE FAILED	The open of the device failed.
WRONG MACHINE NAME	The called machine is reporting a different name than expected.
CALLBACK REQUIRED	The called machine requires that it calls your system.
REMOTE HAS A LCK FILE FOR ME	The remote site has a LCK file for your system. They could be trying to call your machine. If they have an older version of UUCP, the process that was talking to your machine may have failed leaving the LCK file. If they have the new version of UUCP and they are not communicating with your system, then the process that has a LCK file is hung.
REMOTE DOES NOT KNOW ME	The remote machine does not have the node name of your system in its *Systems* file.
REMOTE REJECT AFTER LOGIN	The login used by your system to log in does not agree with what the remote machine was expecting.
REMOTE REJECT, UNKNOWN MESSAGE	The remote machine rejected the communication with your system for an unknown reason. The remote machine may not be running a standard version of UUCP.
STARTUP FAILED	Login succeeded, but initial handshake failed. Check communication parameters: data word size, parity, stop bits, and so on.
CALLER SCRIPT FAILED	This is usually the same as DIAL FAILED. However, if it occurs often, suspect the caller script in the *Dialers* file. Use **uutry** to check.

Chapter 19

Setting up electronic mail

This chapter explains how to set up electronic mail on your SCO UNIX system.

Electronic mail on the SCO UNIX system is handled by two utilities, the MAIL USER AGENT (MUA) and the MAIL TRANSPORT AGENT (MTA). The MUA is the program, such as **mail**(C), that allows users to send, read, and manage mail messages. The MUA transfers the message to the MTA, the group of programs that actually route and deliver messages to their destinations. The MTA on the SCO UNIX operating system is MMDF (**M**ultichannel **M**emorandum **D**istribution **F**acility).

MMDF provides users with transparent access to the different networks and related mail transport PROTOCOLS, through CHANNELS, regardless of the MUA. (A channel is the method, such as UUCP, used to deliver messages; a protocol is a set of rules for communicating over a network and includes standards for mail message formats.)

In addition, MMDF provides the system administrator with tools to monitor and customize MMDF. Using these tools, the system administrator can tune MMDF dynamically, modifying the behavior of these programs even while MMDF is running.

With MMDF, users can send mail on the local network or over larger area networks across the interconnected group of networks known as the "Internet". The Internet is otherwise known as ArpaNet or DARPA (**D**efense **A**dvanced **R**esearch **P**rojects **A**gency) Internet.

The version of SCO MMDF provided with your operating system was derived from MMDF-II Release 43 from the University of Delaware. This version provides additional features, such as a more robust locking mechanism that allows you to specify how to lock user mailboxes, and support for **sendmail** *.forward* files. This version also differs in the format of the hashed table database built by **dbmbuild**(ADM), which is non-standard.

The system automatically configures MMDF for local (one system) mail delivery when you install your operating system. If you did not install the entire distribution, you should install the MAIL package now using the **custom**(ADM) utility. See your *Installation Guide* for information on installing packages with **custom**.

Chapter overview

The first part of this chapter explains how to configure MMDF for most sites using the simple configuration utility provided with your distribution. First, the chapter gives a basic overview of how MMDF works and covers some background information necessary for using the configuration utility. With this information, you can then fill out the MMDF configuration checklist provided. Then, this chapter shows you how to run the configuration utility to set up MMDF to exchange mail with other computers.

The configuration utility does not work for all site configurations, therefore the second part of this chapter describes the different configuration files, how MMDF uses them, and how to modify them to work with your configuration. This section also covers how to test and maintain the MMDF system.

If you do not plan to exchange mail with other machines via UUCP or TCP/IP, you do not have to run the configuration utility; MMDF is already configured to send mail on the local machine. However, if you want to reroute mail sent to special system accounts, such as *root*, the configuration utility provides an easy way for you to set this up. For more information, see the sections on redirecting mail later in this chapter.

How MMDF works

This section gives you an overview of how MMDF processes and delivers mail on your UNIX system. Most of the concepts relate to mail traveling in both directions; you should read both the outgoing and incoming sections with this assumption.

Outgoing mail

Outgoing mail starts when the user invokes the MUA, such as **mail**(C), to compose a mail message. The MUA requires that the user specify a "To:" header when creating the message. Then, the MUA adds two other headers, "Date:" and "From:", when the user sends the message. These headers specify how MMDF sends the message through the system; the next section explains the format of these headers.

Mail headers

For a mail header to be correct, it must include these three lines in the following format:

```
From: sender
To: recipient
Date: Weekday Mon DD hh:mm:ss year
```

Here is an example:

```
From: fred@npr.COM (Fred Astaire)
To: ginger
Date: Wed Apr  3 12:21:23 PST 1991
```

In addition, most MUAs and mail submission programs, such as **mail**(C) and **execmail**, add extra header lines. For example, the MUA might add the "Message-Id" header. If the MUA does not add this header, MMDF adds it, as well as any "Received" headers. The user can also add other headers. For example, if the user specifies a carbon-copy recipient, the message header includes a "Cc:" line. MMDF allows these additional header lines, but does not use them; the recipient mail server and MUA handle these header lines.

In the example above, note the format of the address in the "From:" line. This is an example of a DOMAIN NAME. MMDF uses the domain name to determine how to route the message. Before you can configure MMDF with the configuration utility, you must understand how domain names work; the section on the configuration checklist later in this chapter covers domains and the different types of domain names, including fully-qualified domain names, in greater detail.

Figure 19-1 shows the path that the message takes through MMDF once you send your message with the MUA.

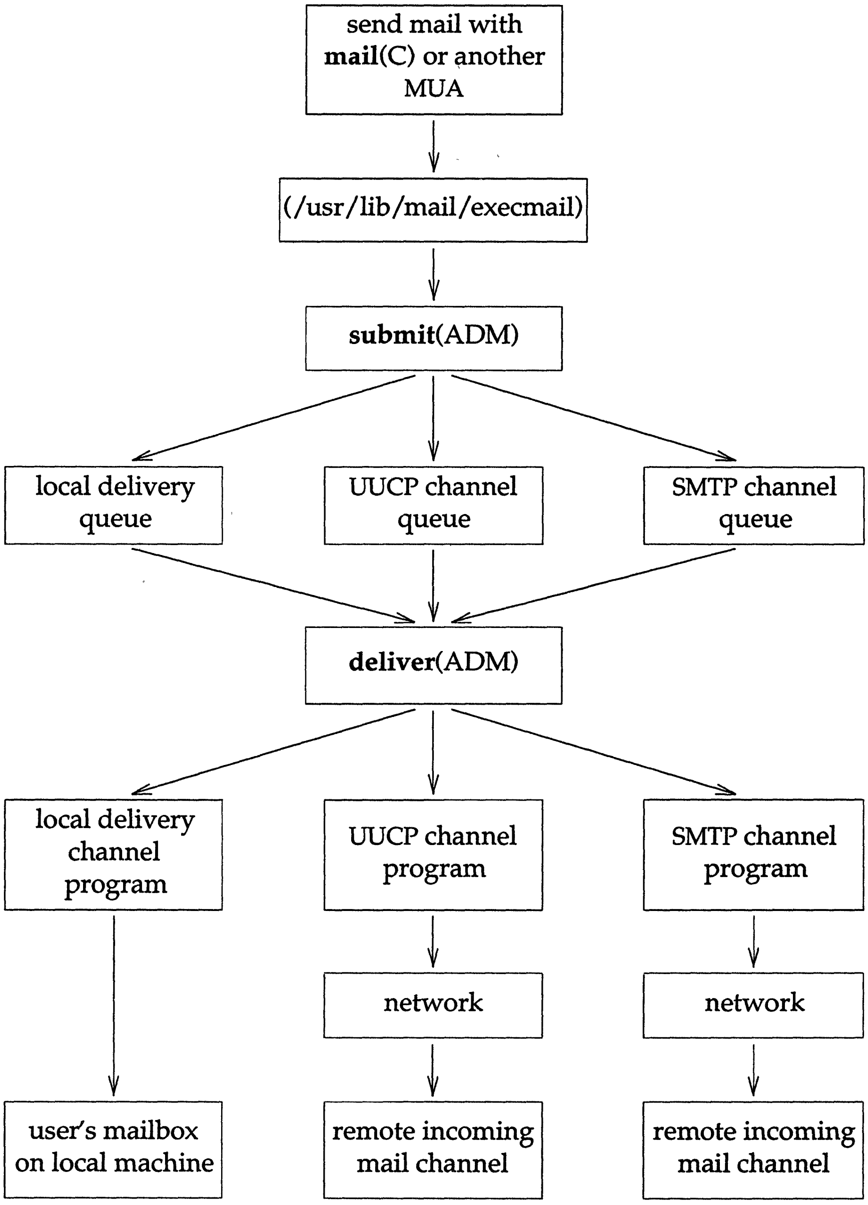


Figure 19-1 Outgoing Mail Route

The MUA allows the user to create, edit, and start a message on its journey. Then, the MUA transfers the message to **execmail**, which sends it to **submit**(ADM). The **submit** program uses the information in the "To:" line and the MMDF configuration files to build the fully-qualified domain name. From this, **submit** determines the channel to use (for example, UUCP) when sending the message and then places the message in the appropriate channel queue for processing later by the **deliver**(ADM) program. The next time **deliver** runs (by default, **deliver** runs every 10 minutes), it transfers the messages from the channel queue to the appropriate channel program.

For example, you send a message with the following "To:" line:

```
To: andrei@npr.com
```

The **submit** program looks in the appropriate MMDF configuration table for *npr* and builds the fully-qualified domain name; for example:

```
scribe.npr.com
```

Now, **submit** determines the appropriate channel to use to send the message. In this case, for example, all messages in the *npr.com* domain are sent to the outside world on the UUCP channel, so **submit** puts the message in the UUCP channel queue. Then **deliver** picks up the message and passes it to the channel program. The channel program transfers the message out of the MMDF mail system and into the UUCP subsystem where it is queued and sent via UUCP.

Incoming mail

MMDF processes incoming mail in much the same way it processes outgoing mail; Figure 19-2 illustrates how MMDF handles incoming mail.

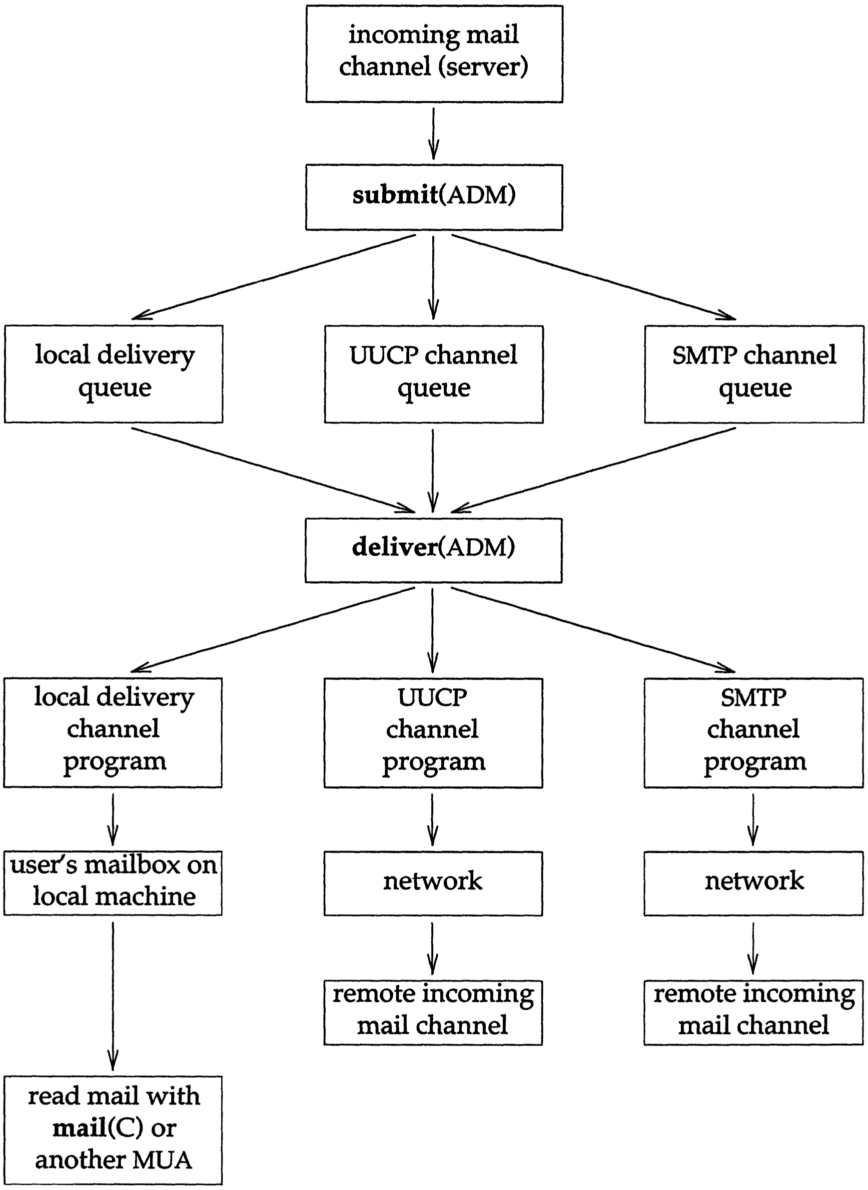

incoming mail channel (server)
submit(ADM)
local delivery queue
UUCP channel queue
SMTP channel queue
deliver(ADM)
local delivery channel program
UUCP channel program
SMTP channel program
user's mailbox on local machine
network
network
remote incoming mail channel
remote incoming mail channel
read mail with **mail**(C) or another MUA

Figure 19-2 Incoming Mail

MMDF processes incoming mail exactly as it processes outgoing mail (the same as when the MUA calls **submit**(ADM)) except that MMDF adds a "Received" line.

MMDF receives incoming mail from a channel's input program (called a "server"), such as **rmail**(C) for the UUCP program. The **submit** program determines the destination of the message and then designates the channel to use to deliver the mail. At this point, MMDF generally selects channels such as **local, badusers,** and **badhosts.** Mail destined for the local machine goes to the **local** channel; when the machine is local, but the user is not, mail goes to **badusers**; and if it requires further processing, the message goes to the **badhosts** channel. If the message is destined for another machine, such as when the local machine is acting as a gateway between networks, MMDF selects another network channel, such as UUCP.

The **submit** program puts the message in the appropriate queue and the **deliver** program transfers the mail from each queue to the appropriate channel; eventually the message is delivered to the user.

The configuration utility

The configuration utility facilitates MMDF configuration by building the appropriate configuration files at installation time. All sites should use the configuration utility to perform the initial MMDF configuration.

The configuration files that this utility builds are sufficient for most sites. If your site is one that the configuration utility does not handle, after building the initial configuration, you must edit the MMDF configuration files manually using the information in the sections "MMDF configuration files" and "Modifying MMDF configuration." (The section "Configuration utility limitations" later in this chapter explains the limitations of this utility.)

In addition, if your configuration changes (for example, you add another machine to your network), you must rebuild the configuration files either by rerunning the configuration utility or by editing these files manually.

Configuring communications channels

Before you set up MMDF, you must configure all the communications channels that you plan to use to route mail. MMDF supports the following channels:

- UUCP (UNIX-to-UNIX Copy subsystem)
- SMTP (TCP/IP)*

NOTE SMTP and TCP/IP channels form part of the TCP/IP product and are not supplied with the UNIX operating system.

The configuration utility configures for all the channels provided by SCO: UUCP, and SMTP. MMDF also supports other third-party mail products that provide channel programs; however, you cannot use the configuration utility to set up MMDF to use these products. See the section "Advanced MMDF configuration tasks" for information on configuring MMDF for use with other channels. For example, if you plan to exchange mail with other hosts via UUCP, you must configure UUCP on the local host before running the configuration utility. For information on configuring UUCP, see the "Building a remote network with UUCP" chapter in this guide.

Configuration utility limitations

This section discusses the situations in which the configuration utility builds either an incorrect or incomplete MMDF configuration for your site. If the situation at your site matches one or more of the situations in this list, you must manually update the MMDF configuration files after building the initial configuration with the configuration utility.

- You use more than one communications channel to exchange mail with a particular host.

  If your host communicates with another host using more than one channel, the configuration utility sets up MMDF to communicate with that host using the first configured channel in the following order:

  1. UUCP
  2. TCP/IP

* The configuration utility performs basic configuration for routing mail over TCP/IP; however, if this configuration is not complete or you change your MMDF configuration, you must refer to your *TCP/IP Administrator's Guide* for details on modifying MMDF configuration.

NOTE If TCP/IP is installed on your system, you must run the configuration utility for UUCP and TCP/IP at the same time. In other words, you should *not* run the configuration utility twice (once for UUCP and once for TCP/IP).

If your host accesses another host using two or more channels, but you do not want to transfer mail using the first applicable channel, the configuration utility does not configure the MMDF correctly. See the "Modifying MMDF configuration" section for more information.

- Your host does not exchange mail with every host on each configured channel.

  The configuration utility automatically sets up MMDF to exchange mail with every host on the configured channels. For information on removing hosts from a configured channel, see the sections on modifying the information for that channel in "Modifying MMDF configuration".

- If you exchange mail with two hosts and you want to prevent one host from passing mail to the other host through the local host.

  The configuration script sets up MMDF to allow one host to send mail through the local host to the other host. If you want to prevent this, you must set up authorization to prevent the transfer. For information on how to do this, see the section "Configuring MMDF authorization" later in this chapter.

- You have more than one channel of the same type.

  The configuration utility lists all the hosts that your host accesses using a particular communications protocol in the same channel. To set up authorization to restrict a host's access on a channel, see "Configuring MMDF authorization."

If your configuration changes for some reason after running the utility (for example, you might add a new UUCP host), you should re-examine this list. If the utility is still able to set up your site correctly, you can update your configuration by running the utility again; otherwise, you must update the configuration files manually.

MMDF configuration checklist

Once you configure the communications channels, use the checklist in this section to write down the information that you need to respond to prompts during the configuration procedure. Items in the checklist that are surrounded by boxes pertain to specific configurations (for example, if you are routing mail over UUCP); fill out these sections only if you have these configurations.

The sections following the checklist contain information on each of the items in the checklist.

Configuration Checklist

1. **Host name:**	____________
2. **Domain name:**	____________
3. **Fully-qualified host name:**	____________
4. **Hide your host name?**	☐ Yes ☐ No
5. **Redirect mail for *root*?:**	____________
6. **Redirect mail for *mmdf*?:**	____________
7. ***postmaster* address:**	____________

If you have UUCP installed and configured and you plan to use MMDF to route mail over UUCP:

UUCP host name:	____________
UUCP hosts with full domain names?	☐ No ☐ Yes; enter domain names: ____________ ____________ ____________ ____________

If you have TCP/IP installed and configured and you plan to route mail over TCP/IP (SMTP):

Configure a name server?	☐ Yes ☐ No

8. **Smart host (badhosts):** (where to route mail to unrecognized machines)	____________
9. **Smart host (badusers):** (where to route mail to unrecognized users)	____________

Use the information in the following sections to complete the checklist.

Host name

The HOST NAME (or machine name) is the name of the machine on which you are configuring MMDF. To determine the host name, enter this command at your UNIX system prompt:

```
cat /etc/systemid
```

An example of a host name is *scribe*.

Domain name

A DOMAIN NAME is the section of a mail address that appears to the right of the at "@" character, for example, *npr.com*. The domain describes the site where your machine is located and generally includes the machine (host) name, a department (optionally), and the site's organization or country. MMDF uses the domain to deliver the message to the appropriate location. Note that the domain name uniquely identifies a machine, but not the path by which messages reach that machine.

The following is the convention for specifying domains:

hostname.subdomain.top-level

If the domain includes a department, the convention is:

hostname.local.subdomain.top-level

Here is a description of each of the domain levels:

Top-Level Domain

A top-level domain is an officially registered name that describes the purpose of a group of institutions or a code that is associated with a country.

You can only use registered top-level and subdomain names if you have registered your organization with SRI International. For information on registering your domain, see the section "Registering domain names" later in this chapter. If you have not registered with SRI, use the UUCP top-level domain.

In the United States, the common top-level domains on the Internet are: *

COM	commercial institutions
code	country code †
EDU	educational and research institutions
GOV	government institutions
MIL	military institutions
NET	network
ORG	organization (generic)
UUCP	an unregistered domain name where users transmit information between cooperating neighbor machines via UUCP.

Subdomain
An officially registered name that describes a company, department, or any subgroup under a top-level domain; *sco* is an example of a subdomain in the domain *COM*.

Local Domain
A name recognized only within an organization that has meaning only within that organization; a department name such as *engr* is an example of a local domain.

Fully-qualified host name

The complete domain name (the machine name and all other domain names) is known as the FULLY-QUALIFIED HOST NAME (or fully-qualified domain name). For example, if you have this mail address:

```
andrei@scribe.npr.com
```

the fully-qualified domain name is:

```
scribe.npr.com
```

In this case, *scribe* is the machine name, *npr* is the company, and *com* specifies that the machine belongs to a commercial organization.

The following table shows examples of other fully-qualified domain names:

* Domain names can be either upper or lowercase; MMDF is case-insensitive when evaluating domain names.

† The International Standards Organization (ISO) standard 3166 defines the country codes. For example, US is the country code for the United States, AU for Australia, DK for Denmark, and JP for Japan.

slug.ucsc.EDU	*slug* is a machine at the University of California at Santa Cruz in the *EDU* domain
seismo.css.GOV	*seismo* is a machine at the Center for Seismographic Studies in the *GOV* domain
nessie.edinburgh.ac.UK	*nessie* is a machine at Edinburgh University in Scotland in the academic (*ac*) subdomain in the *UK* domain

To the configuration utility, the fully-qualified host name is the host name followed by the local (if appropriate), subdomain, and top-level domain name as shown in the previous examples. The domain name refers to just the subdomain and top-level domain name (without the host name). For example, *npr.com.*

Registering domain names

If you or any user at your site plan to receive or send mail outside of your organization, you should register a top-level domain or subdomain with the NIC (Network Information Center). Even if you are only using UUCP, registration with the NIC is your only guarantee that the name of your site is unique.

To register a top-level domain and subdomain name call or write to the NIC standards organization at the following address:

DDN Network Information Center
SRI International
333 Ravenswood Avenue, Room EJ291
Menlo Park, CA 94025 USA
1-800-235-3155

The earlier your site enrolls a domain name and the NIC gives you an address, the less likely it is that you will have to alter a machine name or other site identifier later.

Hide your host name?

If you have several machines at your site (domain) and you register your domain name with the NIC, you can "hide" your host name behind your domain name. When you configure MMDF to hide your host name, people outside your organization can send mail to people who receive mail on different machines within your organization without having to know the name of those specific machines. In this case, MMDF identifies all outgoing mail as coming from a single source (your registered domain name). In addition, MMDF routes mail sent to this source to its correct destination within your organization.

For example, the fully-qualified domain name for the host that you are configuring is:

scribe.npr.com

You can "hide" the hostname *scribe* behind the domain name *npr.com*. In this case, someone outside the *npr* organization can send mail to *andrei* on *scribe* without specifying the fully-qualified domain name. Instead, they can use *andrei@npr.com* as the address.

In addition, when *andrei* sends mail outside the organization, the message appears as if it came from *andrei@npr.com* instead of from *andrei@scribe.npr.com*.

If you want to hide the current host behind the registered domain name, press ⟨Return⟩ when the configuration utility prompts you.

NOTE When you join local machines under a single domain name, you create an administrative domain. Within an administrative domain, all user names must be unique so that mail can go to any person anywhere within the domain without a local machine name in the mail address.

If you decide to hide the host behind a domain name, you should create ALIASES to map each user in the domain to the machine where they receive mail. For information on how to do this, see the section "Creating aliases for users" later in this chapter.

Redirect mail for root?

The system sends mail about any system problems to the *root* user. You can configure MMDF to redirect this mail to another person. For example, if you are the system administrator for a group of machines, you can redirect all mail sent to *root* on those machines to your own system mailbox. With this configuration, you do not need to log into each machine to read *root*'s mail.

If you want to redirect *root*'s mail, press ⟨Return⟩ when prompted by the configuration utility and then enter the address of the person to whom you want MMDF to deliver *root*'s mail. For example, if you are configuring MMDF on *scribe.npr.com* and you want the user *bob@talk.npr.com* to take care of mail to *root*, redirect *root*'s mail to *bob* by entering this at the prompt:

bob@talk.npr.com

(You can redirect *root*'s mail to a local user by entering just the user name at the prompt; you do not need to enter the fully-qualified domain name for a local user.)

Redirect mail for mmdf?

When problems occur with the mail system, the system sends mail to the *mmdf* user. This account is reserved for administering your mail system; unless you log in as *mmdf* regularly, you might not find out about problems with the system. For this reason, it is a good idea to redirect *mmdf*'s mail to another person. To do this, press ⟨Return⟩ when the configuration utility prompts you and then enter the address of the person to whom you want to deliver *mmdf*'s mail.

Postmaster address

On the Internet, people generally send any inquiries about user and host names to the "postmaster" address in that domain. RFC (Request for Comments) 821/822 requires that every host provide this reserved postmaster mailbox. For these reasons, you should designate a user on the local system or within the domain as the postmaster and define a *postmaster* alias. To set up the alias while running the configuration utility, simply enter the name of the postmaster user.

Setting up MMDF for UUCP

If you are setting up the system to exchange mail with another system using UUCP, the configuration utility prompts you for some information about your UUCP hosts.

UUCP host name

The configuration utility prompts for the UUCP name for the host that you are currently configuring. The configuration utility gets this information from **/etc/systemid** (or **uname**); when you install the operating system, the installation prompts you for the system name. In cases where the UUCP name is different from the host name, you should enter the correct name when prompted.

UUCP hosts with full domain names

The configuration utility asks if any of the UUCP hosts have full domain names. This might be the case if your host is not on the Internet but your site has an agreement with another machine on the Internet to transfer your mail. When people on the Internet send messages to you, they use the ***user@machine.domain*** address format (instead of the ***machine!user*** UUCP format). You then use UUCP to connect to that machine and pick up your spooled mail.

For example, your machine *scribe.npr.com* connects to *slug.ucsc.edu* on the Internet. The machine *slug.ucsc.edu* is a UUCP host with a full domain name. In this case, enter **y** when the configuration utility prompts you for UUCP hosts and enter the host name and then the fully-qualified domain name at the prompts.

Configuring MMDF to use a name server

Versions of MMDF included with SCO UNIX System V Release 3.2 Operating System Version 4.0 and later include NAME SERVER support. A name server is a program running on the network that provides a central database of information, such as Internet addresses and the names of hosts on which people receive mail.

If you want the configuration utility to set up MMDF to use the name server, you must set up the name server before running the utility. See Chapter 4, "Name Server Operations Guide for BIND", in your *TCP/IP Administrator's Guide* for details on setting up the name server.

NOTE The name server forms part of the TCP/IP product, not the UNIX operating system.

If you already have a name server running, the configuration utility sets up MMDF to use it automatically.

Configuring smart hosts

If the machine that you are configuring communicates directly with another machine that has more complete information about the entire mail network, you can set up MMDF to route any mail that it does not recognize to that machine. This machine is known as a "smart host". MMDF recognizes two kinds of smart host: hosts with information about machine names and hosts with information about the users on the network.

Badhosts channel

If you specify a smart host for the machine names, MMDF routes mail destined for machines that it does not recognize to the **badhosts** channel. All mail directed to this channel is sent to the smart host that you specify via the normal channel (such as UUCP) that you use to communicate with that host.

For example, someone sends mail to *boris@kgb.gov* but the local host does not recognize the machine *kgb.gov*. However, your host uses UUCP to communicate with another machine, *rocky.npr.com*, that has more complete information about the network. If you configure *rocky.npr.com* as your smart host, MMDF puts the message to *boris@kgb.gov* in the **badhosts** channel and then uses UUCP to deliver it to *rocky.npr.com*. The mail system on *rocky.npr.com* then determines the correct route to send the message to its destination.

Badusers channel

You can also specify a smart host that contains complete information about all the users on a network. This is useful if you have a large number of people at your site. Instead of maintaining information on all the people at the site on each host, you can maintain this information on one central host (the smart host); each individual host maintains information about the local users only. In this case, any mail addressed to users that the local machine does not recognize is directed to the **badusers** channel. The mail routed on this channel is sent to the smart host via the channel (such as UUCP) that you usually use to communicate with that host.

For example, someone on the local host sends mail to *natasha* but the host does not recognize her as a local user. If you communicate with *moose.npr.com*, a smart host that contains complete user information, MMDF routes the message to this host. This does not mean that *natasha* actually receives her mail on *moose*, just that *moose* has more information about where *natasha* is located.

Name server setup

Before running **mkdev mmdf** check that your name server is an authoritative domain. Refer to your TCP/IP configuration documentation for further details.

Running the configuration utility

Once you configure the communications channels (and name server, if appropriate) and complete the configuration checklist, you are ready to run the MMDF configuration utility.*

NOTE The configuration utility provides common default values for many of the prompts; to accept these values, press ⟨Return⟩ at the prompt.

The configuration procedure varies slightly (from step 13 onwards) depending on whether TCP/IP is installed. The procedure detailed in this section assumes that TCP/IP is not installed. If TCP/IP is installed on your system, follow the instructions down to step 13 and then continue from the section headed "Running the configuration utility with TCP/IP installed" later in this chapter.

Use the following procedure to configure MMDF:

* Remember, if you are only planning to route mail on the local machine, you do not have to run the configuration utility; MMDF is already set up for local mail delivery.

1. If the *mmdf* account does not already have a password, assign a password now (the system does not set the password for *mmdf* at installation time). Enter the following command as *root*:

   passwd mmdf

2. Log in as *mmdf*.
3. Change to the */usr/mmdf* directory and start the configuration utility by entering the following command as *mmdf*:

   mkdev mmdf

   You see a screen with information about the limitations of the configuration utility.

4. At the bottom of the screen, you see this prompt:

```
Do you wish to continue the configuration process at this time? [y]
```

   If you enter **n** at this prompt, you exit the configuration utility and return to the UNIX system prompt. Press ⟨Return⟩ or enter **y** to continue with the configuration.

   The configuration utility displays a message about the current version of the software, for example:

```
This machine is running level 43NS MMDF.
```

5. If you have not already set the host name on the machine that you are configuring, you see the following prompt (if you have already set the host name, skip this step):

```
The name of this host has not been configured yet.
What will you be calling this host?
```

   Enter the host name at this prompt. You must set the name of the host before configuring MMDF.

6. Now, you see the following prompt where 'host.domain' is a fully-qualified host name like *scribe.npr.com*:

```
Is your fully qualified host name 'host.domain'? [y]
```

If the fully-qualified host name is correct, press ⟨Return⟩ and go to the next step. If the host name is incorrect, enter **n**. The utility displays this prompt:

```
What is the correct host name? [host]
```

Enter the correct hostname (for example, *pubsco*) or press ⟨Return⟩ to select the default hostname. You see the prompt for the domain name:

```
What is the correct domain name? [domain]
```

There are three possible responses to this prompt: you can press ⟨Return⟩ to accept the default domain name, you can enter a different domain name (for example, *eng*), or you can enter a domain and sub-domain name (for example, *techpubs.eng*).

Now, the utility prompts you to confirm the fully-qualified host name:

```
Is your fully qualified host name 'pubsco.techpubs.eng'? [y]
```

If the hostname is correct, press ⟨Return⟩ and go to the next step; otherwise enter **n** and repeat this step.

If you have specified a fully qualified host name which includes a sub-domain, for example, *pubsco.techpubs.eng*, the prompt shown in step 7 is displayed; if you have specified a fully qualified host name which does not include a sub-domain, for example, *pubsco.eng*, step 7 is bypassed.

7. Now, you see some information about "hiding" the host behind the domain, followed by this prompt:

```
At many sites, it is common for mail to be addressed as being from
"person@site" instead of from "person@machine.site". This allows people
to be moved between machines internally without requiring them to notify
all their external correspondents about the address change. This
configuration does, however, require a complete user alias table (see
below) containing mappings from user names to the host that they actually
plan to read their mail on. If you are not sure, then you should
probably answer "yes" if there are two or more machines in the domain
"techpubs.eng", and "no" otherwise.

Do you wish to hide 'pubsco.techpubs.eng' behind 'techpubs.eng'? [y]
```

To "hide" the host that you are configuring behind another host, press ⟨Return⟩ at this prompt. Otherwise, enter **n**.

8. Now, the configuration utility displays some information about creating mail aliases for special accounts, *root, mmdf,* and *postmaster,* on your system.

   First, the utility prompts you to create an alias for *root*:

```
Do you wish to have mail for root redirected to a real user? [y]
```

   If you want mail addressed to *root* to go to *root*'s system mailbox, enter **n** and go to the next step.

   If you want *root*'s mail to go to a different person, press ⟨Return⟩ and you see this prompt:

```
To whom should root's mail be sent?
```

   Enter the login name of the person that you want to receive mail directed to *root.*

9. Now, you see this prompt:

```
Do you wish to redirect mail addressed to mmdf? [y]
```

   If you do not want to redirect mail to *mmdf,* enter **n** and go to the next step. If you want mail addressed to *mmdf* to go to another person (for example, the system administrator), press ⟨Return⟩.

```
To whom should mmdf's mail be sent? [root]
```

   The default for this prompt is the same as the selection made for step 8. For example, if *root* was selected in response to step 8 then *root* is displayed here.

   To have *mmdf*'s mail go to *root*'s mailbox, simply press ⟨Return⟩. If you want this mail directed to another person, enter the address of that person.

10. You see this prompt:

```
To whom should mail addressed to postmaster be sent? [mmdf]
```

    Again, you can direct any mail sent to *postmaster* to another person, such as *mmdf*. Press ⟨Return⟩ to accept the default or enter a different address.

11. If UUCP is not installed on your system, you see a message like this:

```
UUCP not installed, skipping ...
```

12. Now, you see some information about associating login names for each user to the machines where the users actually read their mail by creating an alias file. See the section "Creating aliases for users" later in this chapter.

    The configuration utility also displays some information about converting XENIX aliases to the correct MMDF format. For more information, see the section "Converting XENIX alias files" later in this chapter.

    If you have UUCP installed on your system, you see this prompt:

```
Are you going to be using UUCP for mail? [y]
```

    If you plan to use UUCP for transferring mail, press ⟨Return⟩.

    NOTE If you are running the configuration utility with TCP/IP installed on your system, go to the section "Running the configuration utility with TCP/IP installed."

13. Now, you see the prompt for the UUCP host name:

    ```
    Is this host known as 'host' for UUCP? [y]
    ```

    If the host name is correct, press ⟨Return⟩ and go to the next step. Otherwise, enter **n** to display the following prompt:

    ```
    What is this host's UUCP name? [host]
    ```

    Enter the new name and press ⟨Return⟩. You see this prompt again:

    ```
    Is this host known as 'host' for UUCP? [y]
    ```

    Press ⟨Return⟩ if the host name is correct, and go to the next step, or press **n** to change it.

14. You see this prompt:

    ```
    UUCP must be configured before mail
    ```

    In this case, configure your UUCP connections before running **mkdev mmdf** again.

15. Now, the configuration utility prompts you for information about the hosts you communicate with using UUCP:

```
Because UUCP does not maintain information about domain names, it will be
necessary for you to provide the domain names of any of the hosts with
which you communicate via UUCP.

Do any of your UUCP hosts have full domain names? [n]
```

    In most cases, UUCP hosts do not have full domain names, so you can simply press ⟨Return⟩ at this prompt.

    If any of your UUCP hosts have full domain names, enter **y**. You see the prompts for the host names of your UUCP hosts:

```
Enter the UUCP site name (blank to terminate):
```

    Enter the host name, for example, *pubsco* (not the fully qualified name) and press ⟨Return⟩.

16. The following screen is displayed:

```
What is pubsco's fully qualified name? [pubsco]
```

    Enter the fully qualified name, for example, *pubsco.techpubs.eng* and press ⟨Return⟩.

17. You see the following messages:

```
Micnet not configured, skipping ...

TCP/IP not installed, skipping ...

Many sites do not have complete information about the entire mail network,
but rely on another "smarter" host to determine the correct route that
mail messages should follow to reach their destinations. Any mail that
the local machine is incapable of correctly handling is passed to the
smart host for further processing. In MMDF, this is called the "badhost"
channel.

Do you have such a "smart" host? [y]
```

    If you enter **n**, go to the next step; if you enter **y** the following screen is displayed:

```
What is its name? (q if you have changed your mind)
```

18. The following screen is displayed:

```
Another option, which is often used on large sites is to have a central
machine which contains complete knowledge about all the users on the site,
and only maintaining local lists on each machine. MMDF provides the
facility to forward mail containing unrecognised local addresses to a
smarter host which will have a complete user data base (via the "baduser"
channel).

Do you have such a "smart" host? [y]
```

If you enter **n**, go to the next step; if you enter **y** the following screen is displayed:

```
What is its name? (q if you have changed your mind) [pubsco.techpubs.eng]
```

19. When you finish, the configuration utility displays information like the following as it creates and edits the MMDF configuration files (the messages might be different, depending on your configuration):

```
Creating the mmdftailor file: header, host name info, support
address, alias tables, local domain and channel, SMTP, root

Building the alias tables (mostly empty)
A Mailing list alias table (alias.list) already exists, skipping
A general user alias table (alias.user) already exists, skipping

Building channel files
local, list, SMTP, UUCP, badusers, badhosts

Building the domain tables
local, domain, root
done
building the database
```

The configuration is now complete.

Running the configuration utility with TCP/IP installed

This section assumes that TCP/IP is installed on your system. It continues on from step 12 in the previous section.

13. You see the following messages:

```
Micnet not configured, skipping ...

Are you going to be using SMTP for mail? [y]
```

If you enter **n** go to step 15; if you enter **y** go to step 14.

14. The configuration utility displays one of the following messages:

- If the name server is already configured on your system, you see the following screen:

```
A domain name server is running, mail will be configured to use it.
```

In this case, the configuration utility configures MMDF on the host that you are configuring to use the name server automatically.

- If the name server is not configured, you see:

```
The name server is not currently configured on this machine.
Do you plan to configure a name server on the local network? [n]
```

If you do not want to configure a name server, press ⟨Return⟩ to continue with the configuration utility.

If you plan to configure a name server, you must do so before configuring MMDF. In this case, enter **y**. The configuration utility exits and displays this message:

```
Name server must be configured before mail.
```

Set up the name server and then run **mkdev mmdf** again. For information on configuring the name server tables, refer to Chapter 4, "Name Server Operations Guide for BIND," in your *TCP/IP Administrator's Guide.*

15. The following screen is displayed:

```
Many sites do not have complete information about the entire mail network,
but rely on another "smarter" host to determine the correct route that
mail messages should follow to reach their destinations. Any mail that
the local machine is incapable of correctly handling is passed to the
smart host for further processing. In MMDF, this is called the "badhost"
channel.

Do you have such a "smart" host? [y]
```

If you enter **n**, go to the next step; if you enter **y** the following screen is displayed:

```
What is its name? (q if you have changed your mind)  [ ]
```

16. The following screen is displayed:

```
Another option, which is often used on large sites is to have a central
machine which contains complete knowledge about all the users on the site,
and only maintaining local lists on each machine. MMDF provides the
facility to forward mail containing unrecognised local addresses to a
smarter host which will have a complete user data base (via the "baduser"
channel).

Do you have such a "smart" host? [y]
```

If you enter **y** the following screen is displayed:

```
What is its name? (q if you have changed your mind)  [ ]
```

17. When you finish, the configuration utility displays information like the following as it creates and edits the MMDF configuration files (the messages might be different, depending on your configuration):

```
Creating the mmdftailor file: header, host name info, support
address, alias tables, local domain and channel, SMTP, root

Building the alias tables (mostly empty)
A Mailing list alias table (alias.list) already exists, skipping
A general user alias table (alias.user) already exists, skipping

Building channel files
local, list, SMTP, UUCP, badusers, badhosts

Building the domain tables
local, domain, root
done
building the database
```

The configuration is now complete.

Preparing MMDF for use

Before you can use your mail system, you must perform steps to notify the UNIX system of the new configuration. The following sections contain information on restarting **deliver** and making this change permanent in the MMDF system startup file.

Restarting the deliver daemon

When you enter multiuser mode, the system automatically starts a **deliver** process for the local channel only. Each time someone on the system sends or receives a message, the **deliver** daemon runs, placing the message in the correct channel.

With the MMDF configuration utility, you configured MMDF to use additional channels. However, the configuration utility does not affect the **deliver** daemons that are currently running. To make the changes to your MMDF configuration take effect, stop and restart any **deliver** daemons using these commands:

1. As *root,* enter the following command:

   ps -ummdf | grep deliver

   This command displays any **deliver** processes running on your system. For example:

```
285  ?        0:00  deliver
```

2. Stop the **deliver** process with **kill**(C). For example:

   kill 285

3. Now, restart **deliver** with this command:

   /usr/mmdf/bin/deliver -b

   This command restarts the **deliver** daemon so that it runs every 10 minutes. If you want **deliver** to run more often, use the **-T** option. We recommend that you set up **deliver** to run every 60 seconds using the following command:

   /usr/mmdf/bin/deliver -b -T60

NOTE When you restart the **deliver** daemon, the change is only temporary. The next time you reboot, the system reads the original **deliver** command in */etc/rc2.d/S86mmdf* and starts **deliver** for the local channel only. To make your changes permanent, modify the *S86mmdf* file using the information in the next section.

Modifying MMDF system startup

Unless you want to run multiple **deliver**(ADM) daemons, you do not need to modify MMDF system startup. The MMDF system startup file */etc/rc2.d/S86mmdf* already includes information for starting the **deliver**(ADM) daemon on all the configured channels. The system reads the *S86mmdf* file automatically when you enter **init** state 2 (multiuser mode).

By default, the line in this file looks like the following:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b"
```

To edit *S86mmdf*:

1. First, log in as *root*.
2. Edit */etc/rc2.d/S86mmdf*.
3. If you want **deliver** to run more often than the default of every 10 minutes, add the **-T** option on this line. For example, set up your system so that **deliver** runs every 60 seconds using the following **deliver** startup line:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -T60"
```

NOTE When you start the **deliver** program with the **-b** option only, *one* **deliver** process manages *all* the channels that you specified with the configuration utility. If you want different **deliver** processes to manage each of your channels or more than one **deliver** process for a single channel, refer to the section "Running multiple deliver daemons" later in this chapter.

Other common configuration tasks

Depending on your configuration, you might need to perform some additional MMDF configuration tasks after running the configuration utility and restarting **deliver**. This section covers some common configuration tasks; for additional configuration information, refer to the sections "Modifying MMDF configuration" and "Advanced MMDF configuration tasks" later in this chapter.

Creating aliases for users

If you chose to hide your local host name behind your domain name in step 6, you should add the login names of the users on the system and machine names where they receive mail to the */usr/mmdf/table/alias.user* file. The configuration utility creates *alias.user* for you; add information to this file using the following format:

user: *user@machine.domain*

For example, to map *andrei* to the machine *scribe* in the *npr.com* domain, use this format:

andrei: andrei@scribe.npr.com

See the sections "Alias files" and "Maintaining user aliases" for more information on setting up aliases on your system.

Converting XENIX alias files

If you are converting your mail system from XENIX to UNIX, you must use the **mmdfalias** utility to convert a XENIX-style */usr/lib/mail/aliases* file to the correct format for MMDF. For information on how to do this, see the section "Converting from XENIX" later in this chapter.

Changing logging levels

By default, the configuration utility sets the logging levels to log fatal errors only (FAT). If you are switching to MMDF from a **sendmail**-based system, you might want to change the **MMSGLOG** tunable parameter, in the */usr/mmdf/mmdftailor* file, from FAT to FST to retain a logging level equivalent to "mqueue/syslog". See the sections "Changing error logging levels" and "Changing MMDF parameters" for more information on changing logging parameters.

Rebuilding the MMDF hashed database

The configuration utility rebuilds the MMDF hashed database automatically after modifying the configuration files. However, if you modify the alias files or */usr/mmdf/mmdftailor* **after** running the MMDF configuration utility, you must rebuild this database manually.

To do this, enter the following commands as user *mmdf*:

```
cd /usr/mmdf/table
./dbmbuild
```

NOTE You must rebuild the MMDF database every time you make changes to the alias or routing files in the */usr/mmdf/table* directory.

MMDF configuration files

The configuration utility provided with your MMDF distribution builds configuration files sufficient for most sites, but not for every site; see the section on the limitations of the configuration utility earlier in this chapter for more information. In cases where the files that the configuration utility builds are insufficient, you must edit these files manually. This section describes the different configuration files, how MMDF uses them, and how to modify them to work with your configuration. This section also explains how to test and maintain the MMDF system.

The material presented in this section assumes that you have already set up the channels over which you plan to route mail and that you built the initial MMDF configuration using the configuration utility.

This section begins with an introduction to the different configuration files and formats that MMDF uses to route mail.

Table 19.1 briefly describes the MMDF configuration files.

Table 19-1 MMDF configuration files

File(s)	Description
mmdftailor	defines all the mail attributes for the local machine
alias files	defines aliases for mailing lists, programs, users, and system accounts
domain files	matches host names to fully qualified host names
channel files	expands fully qualified host names to specify the addressing information necessary to reach the host or domain

The following sections describe the format of these files in detail. The **tables**(F) manual page also contains information about the file formats.

The mmdftailor file

The */usr/mmdf/mmdftailor* file defines all the mail attributes for the local machine, such as its name, the domain, channel, and alias files to use, how to set up each channel, and how to perform logging.

By default, MMDF is distributed with a simple *mmdftailor* file that is configured for local mail only. When you perform the initial MMDF configuration (as described in the section "Running the configuration utility"), the configuration utility modifies the organization of the default *mmdftailor* file. This section describes the *mmdftailor* file generated by the configuration utility. Your *mmdftailor* file might be set up differently (for example, the names of your hosts are different and your *mmdftailor* file might not include all the channels in the example).

Table 19.2 briefly describes the keywords in the *mmdftailor* file; the sections that follow the table explain these keywords in more detail.

Table 19-2 mmdftailor file keywords

Keyword	Description
MLDOMAIN	defines the top-level domain name (for example, *COM* or *GOV*)
MLNAME	specifies the host name (*scribe*) or site name (*npr*) for hidden hosts
MLOCMACHINE	specifies the host name (for hidden hosts)
UUname	specifies the host name for UUCP
MSUPPORT	sets the address (*postmaster*) to send undeliverable mail and requests for information
MTBL	associates an abbreviated name with the more descriptive name for the alias, channel and domain files (also called "tables"). These abbreviated names are then used throughout the mmdftailor file as shorthand to refer to the table files.
ALIAS	defines the various sources for alias information, using the abbreviated names specified in the **MTBL** definitions
MCHN	defines the channels available to MMDF for mail transport*
MDMN	describes the domains known to MMDF
MMSGLOG	controls logging information from **deliver**(ADM) and **submit**(ADM)
MCHANLOG	controls logging that **MMSGLOG** does not control
MLCKTYPE	specifies the mailbox locking protocol

The **MTBL, ALIAS, MCHN,** and **MDMN** keywords accept specific parameters; Table 19.3 briefly describes the parameters in the default *mmdftailor* file. See the **mmdftailor**(F) manual page for complete information about each of these keywords and their associated parameters.

* MMDF requires at least one channel (**local**), for delivering local mail. See the section "Local mail configuration" for more information.

Table 19-3 mmdftailor file parameters

Keyword	Parameter	Description
MTBL	name	specifies the shortname (table) for file
	file	describes the filename to associate with the shortname
	show	sets the description to display with programs, such as **checkup**(ADM)
ALIAS	table	specifies the name of the table to associate with the alias entry
	nobypass	prevents the ~***address*** alias bypass mechanism from working on the aliases in the specified table
	trusted	states that any alias in the described file is permitted to deliver mail to files or pipes using the permissions of any user on the system (including *root*)*
MCHN	show	sets the description to display with programs
	ap	specifies the type of address parsing used for the header of outgoing messages
	name	specifies the name of the channel
	mod	sets the delivery mode for the channel
	host	specifies the name of the host that is being contacted by this channel
	tbl	defines the file that lists the hosts that are accessible via the specific channel
	confstr	specifies channel-specific configuration parameters
	pgm	defines the channel program to invoke for this channel
MDMN	show	sets the description to display with programs
	table	specifies the name of the table containing information that describes the sites in this domain
	dmn	specifies the name of the domain

* Normally, because this permits such accessibility, you should only set the *alias* file (used for administrative aliases) to **trusted.**

MLDOMAIN

The **MLDOMAIN** keyword describes the top-level domain used by the local host. If the local machine has a registered top-level or subdomain name, the top-level domain name for your site, such as *COM* or *EDU*, appears in the **MLDOMAIN** line. If your host does not have a top-level domain, use UUCP. (UUCP is a generic name that MMDF uses for UUCP connections.) The **MLDOMAIN** line in the following example shows *COM* as the top-level domain:

```
MLDOMAIN com
```

MLNAME and MLOCMACHINE

Generally, **MLNAME** describes the name of the local host. In the following example, *scribe* is the host name:

```
MLNAME scribe
```

However, if you are "hiding" the local machine behind a domain name, **MLNAME** describes the subdomain name of your site (for example, *npr*) and **MLOCMACHINE** describes the name of the local host. (**MLOCMACHINE** is only used if you are hiding the local host.)

For example, if you are hiding the local host, *scribe,* behind the *npr* subdomain, these lines look like the following:

```
MLNAME npr
MLOCMACHINE scribe
```

For more information about hiding the host name, see the section "Hide your host name?" earlier in this chapter.

UUname

The **UUname** line describes the name used with UUCP; this definition must exist for UUCP to work properly. In most cases, this name is the same as the host name (in either **MLNAME** or **MLOCMACHINE**), but can be different if required. For example:

```
UUname scribe
```

MSUPPORT

The **MSUPPORT** line describes the address to send mail delivery problem notifications. If your site is connected to the Internet, you must define **MSUPPORT** as *postmaster*. On the Internet, people use this address to send any inquiries about user and host names in the domain. In addition, RFC821/822 (Request for Comments, an Internet standard) requires that every host on the Internet provide the reserved *postmaster* mailbox. Use the following line:

```
MSUPPORT postmaster
```

Note that the address you specify with **MSUPPORT** must be legal; if it is not and MMDF cannot deliver the original undeliverable mail to the support address, MMDF creates a new piece of mail that is undeliverable, and so on until the machine runs out of processes.

You must create an alias in the *alias.ali* file to redirect *postmaster*'s mail to a user on the system. For more information, see the "Alias files" and "Changing the postmaster alias" sections later in this chapter.

Alias configuration

The **MTBL** alias configuration keywords identify the filenames for the default alias files. By default, these **MTBL** lines look like the following:

```
MTBL name=alias,    file="alias.ali",    show="Administrative aliases"
MTBL name=lalias,   file="alias.list",   show="Mailing list aliases"
MTBL name=auser,    file="alias.user",   show="General user aliases"
```

Each **name** parameter defines the short name for the actual filename containing alias information (specified by **file**). Thus, the first **MTBL** entry identifies "alias" as the shortname for the *alias.ali* file. (MMDF uses this file to define the system administrative aliases for the local host.) See the "Alias files" section later in this chapter for more information about these files.

The **ALIAS** alias configuration keywords define additional information about the alias files in the **MTBL** lines. The following example shows the default **ALIAS** lines:

```
ALIAS   table=alias,    nobypass,   trusted
ALIAS   table=lalias,   nobypass
ALIAS   table=auser
```

NOTE The order of the alias files defined in this section determines the order that MMDF searches the files to find an alias.

For more information, see the "Alias files" section later in this chapter.

Local mail configuration

As with alias configuration, the **MTBL** keywords identify the filenames that contain local mail configuration information. The following example shows these table definition lines:

```
MTBL name=local,    file="local.chn",   show="Local Host Aliases"
MTBL name=locdom,   file="local.dom",   show="Local Domain"
```

In this example, the first **MTBL** entry identifies "local" as the shortname for the *local.chn* file. (The *local.chn* file contains information about the **local** channel on the local machine.) See the section "Channel files" later in this chapter for more information about these files.

Below the **MTBL** definitions are the **MCHN** and **MDMN** definitions for local mail delivery. By default, the **MCHN** line for local delivery looks like the following:

```
MCHN   local,   show="Local Delivery",   ap=822,   mod=imm
```

The first parameter after **MCHN** is an arbitrary name that describes the channel (in this case, **local**). For a description of the **MCHN** parameters, see the **mmdftailor**(F) manual page. MMDF searches the channel tables in the order that the **MCHN** definitions appear in *mmdftailor*.

The local mail **MDMN** line looks like the following:

```
MDMN   "npr.COM",   show="Local domain",   table=locdom
```

The first parameter on the **MDMN** line specifies the name of the domain that the table lists (in this case, the local domain *npr.COM*).

List processing configuration

The list processing configuration section of *mmdftailor* defines the table and channel that MMDF uses to process mailing lists. By default, these lines look like the following example:

```
MTBL   list,   file="list.chn",         show="List Channel"
MCHN   list,   show="List Processing",  ap=same,   mod=imm,
       host="scribe.npr.COM"
```

The **MTBL** line defines the shortname for the file *list.chn*. This file contains information about passing mail addressed to mailing lists to the **list-processor** program (see the "The alias.list file" section later in this chapter for more information).

SMTP configuration

If you configured your system to route mail over TCP/IP without a name server, your *mmdftailor* file contains **MTBL** and **MCHN** entries like the following:

```
MTBL   smtpchn, file=smtp.chn, show="SCO SMTP Channel"
MCHN   smtp, show="SCO SMTP Delivery", ap=822, tbl=smtpchn, mod=imm,
       confstr="scribe.npr.COM"
```

MTBL describes the SMTP channel file, *smtp.chn*. If you have configured the name server, the MTBL line looks like:

```
MTBL smtpchn, flags=ns,        show="SCO SMTP Channel",
        flags=channel
```

There is also an MCHN entry for the 'delay' channel, of the form:

```
MCHN delay, show="Name server Delay Channel", ap=same,
        tbl=smtpchn
```

The delay channel is used to hold mail until the name server is capable of responding to enquiries made to it. It should usually not contain any mail. Messages waiting in the delay channel for a long time are generally an indication that something is wrong with either the name server configuration or the name server-specific configuration information in MMDF.

If you have messages waiting in the delay channel, check your name server is an Authoritative domain, otherwise *mmdf* will not function correctly.

For more information about parameters available in an MCHN declaration, see the **mmdftailor**(F) manual page.

Local domain table configuration

The **MTBL** and **MDMN** entries in the local domain table configuration section describe the machines in the local domain. These lines look like the following:

```
MTBL domain,    file=domain.dom,       show="Local Ethernet", flags=partial
MDMN "npr.COM", show="Local Ethernet", table=domain
```

The first parameter on the **MDMN** line specifies the name of the domain that the table lists (in this case, the *npr.COM* domain). The *domain.dom* file describes the machines located in the local domain.

UUCP configuration

If you configure MMDF to route mail over UUCP, the configuration utility adds parameters like the following that describe the UUCP configuration information:

```
MTBL uuchn,  file="uucp.chn",  show="NPR UUCP Channel"
MTBL uudom,  file="uucp.dom",  show="NPR UUCP Domain"

MCHN uucp,  show="NPR UUCP Delivery",  tbl=uuchn, ap=same
MDMN "UUCP",  show="UUCP Domain",  table=uudom
```

The two **MTBL** entries define the UUCP channel (*uucp.chn*) and UUCP domain (*uucp.dom*) files. The **MCHN** line describes more information about the UUCP channel and **MDMN** defines the file that describes the machine accesses via UUCP.

The badhosts channel

If you specified a "smart host" to redirect mail destined for machines that the local host does not recognize, the **MCHN badhosts** line looks like this:

```
MCHN badhosts, show="Last-Chance Routing", pgm=smtp, tbl=smtpchn,
     ap=822, host="rocky.npr.COM", confstr=scribe.npr.COM
```

The **MCHN host** parameter describes the smart host (in this case, *rocky.npr.COM*). In this case, the local host accesses the smart host via SMTP. The **pgm** parameter specifies the filename of the channel program (*smtp*) in the */usr/mmdf/chans*) directory and **tbl** specifies the file (*smtp.chn*) that describes the machines that the host accesses with SMTP.

The **badhosts** is not really a channel because it is not associated with its own transport program. In this example, the pseudo-channel uses the SMTP channel to relay mail to a more intelligent host. If the **badhosts** channel does not exist, MMDF returns mail to an unknown host to the sender.

The badusers channel

If you specified a "smart host" to redirect mail to users that the local host does not recognize, the **MCHN badusers** line looks like this:

```
MCHN badusers, show="Last-Chance Routing", pgm=smtp, tbl=smtpchn,
     ap=822, host="moose.npr.COM", confstr=scribe.npr.COM
```

The **MCHN host** parameter describes the smart host (in this case, *moose.npr.COM*). See the previous section for information about the **pgm** and **tbl** parameters.

The root domain table

The **MTBL** line defines the *root.dom* file which contains any domain information not named in other domain files.

```
MTBL rootdom, file="root.dom", show="Root Domain", flags=route
MDMN "", show="Root Domain", table=rootdom
```

The **MDMN** entry defines the name of root domain. (The root domain definition has no name ("") because the root domain file (*root.dom*) can contain entries for many different domains.)

Logging levels

By default, the configuration script sets **MMSGLOG** and **MCHANLOG** to the FAT logging level; this level logs fatal errors only. These entries look like the following:

```
MMSGLOG  level=FAT
MCHANLOG level=FAT
```

You can change these logging parameters to a higher level, such as FST to log full statistics, or add different logging controls, such as **AUTHLOG** to control authorization information. See the section on **MCHANLOG** in the **mmdftailor**(F) manual page for a list of levels. "Changing MMDF parameters" later in this chapter also contains information about these logging levels. For more information about the MMDF logging files, see the **logs**(F) manual page.

Mailbox locking style

By default, MMDF uses the standard System V **fcntl**() kernel file locking protocol to lock users' mailboxes. However, if users on the system use MUAs that do not use the default locking protocol, you can configure the locking type with the **MLCKTYPE** keyword. The default **MLCKTYPE** line looks like the following:

```
MLCKTYPE advisory
```

See the section "Changing MMDF parameters" later in this chapter for more information on setting locking protocols.

Alias files

An ALIAS is an abbreviated name that MMDF translates into a larger string (a mail address or list of addresses). Aliases are useful for specifying a single name to represent a group of users. You might want to create an alias called *sales* to represent all the members of the Sales department in the company. To do this, you define an alias file in the */usr/mmdf/mmdftailor* file and then create the alias file in */usr/mmdf/table*. In the alias file, the *sales* alias entry might look like the following:[1]

```
sales:  joe, jane, bob, mike, karen, ann
```

When users want to send mail to everybody in the Sales department, they can use the *sales* alias like this:

mail sales

This sends a copy of the message to everyone in the Sales department.

You can name your alias files anything you like; however, you must define the file names in the */usr/mmdf/mmdftailor* file (see the "The mmdftailor file" section earlier in this chapter). By default, MMDF provides the files listed in Table 19.4 that you can use to specify aliases for user names:

Table 19-4 MMDF alias files

File	Description
alias.list	aliases for lists of users
alias.user	aliases mapping users to their "home" host machines (the machines on which they receive their mail) and nicknames for local users
alias.ali	local machine system administrative aliases, programs, or files. At the minimum, this file should identify the aliases for *mmdf* and *postmaster*.

These files are located in the */usr/mmdf/table* directory.

1. The colon ":" following the alias name is optional.

NOTE You should **not** create aliases for remote users (users not located at the local site) in the alias files. In other words, do not create an alias like the following, where *moocow.uucp* is not a local machine at your site:

```
david: david@moocow.uucp
```

In this case, if a remote user sends mail to *david@npr.com,* the message is delivered to *david@moocow.uucp.*

In general, users should set private remote-user aliases using their MUA (for example, use the **alias** command in **$HOME/***.mailrc*). For more information, about creating private aliases, see the chapter on electronic mail in the *User's Guide.*

The alias.list file

Use the *alias.list* file to create multiple-user aliases. With list aliases, MMDF processes the mail using the **list** channel and it appears as being from the sender instead of from the first person in the "To:" line.

To use the **list** channel, you must specify the following three lines for each alias:

- the name of the list
- the "*name*-outbound" line that contains the logins that comprise the list
- the "*name*-request" line that provides a login of the list maintainer who makes additions and deletions to and from the list

The following example shows how to use these three lines:

```
writers:            writers-outbound@list-processor
writers-outbound:   hanna,dianna,george,laurie,meg,naomi,steve
writers-request:    hanna
sales:              sales-outbound@list-processor
sales-outbound:     joe,jane,mike,karen,ann,uksales
sales-request:      joe
```

The **list-processor** keyword is a reserved word in MMDF and indicates access to the **list** channel for processing mailing lists. The *sales-request* alias provides a way to request additions or deletions to the *sales* alias.

When defining an alias that contains many user names, you can use a backslash character "\" as a line-continuation character. Use quotation marks ("") to delimit a string containing spaces or punctuation. When using an alias to define another alias, be careful not to create an alias loop.

You can also use the output redirection symbol " > " with the pipe character " | " to do more complex processing such as redirecting messages to files. For example:

```
Loguucp:        "network//usr/spool/log/uucp"
Logmlog:        "network|cat -v >>/usr/spool/log/mlog"
printer2:       "network|/usr/bin/lp -dprinter2"
```

In this example, MMDF pipes mail addressed to *Logmlog* to the **cat**(C) command to log the mail in the *mlog* file. MMDF pipes mail addressed to *printer2* to the **lp**(C) command for printing. These redirection alias examples use the user and group IDs of the user *network*. Although *network* is appropriate in most cases, you can specify any user named in the */etc/passwd* file on your system.

If you have a long list of names for an alias, you might want to include them in another file (instead of listing them directly in the *alias.list* file). To do this, use the **:include** keyword. For example:

```
staff:          staff-outbound@list-processor
staff-outbound: ":include:/etc/alias/staff"
staff-request:  ross
```

In this example, the **:include:** line specifies that you want to use the names listed in the */etc/alias/staff* file to define the alias. Note that you cannot use the backslash character " \ " as a line-continuation character for lists of names in an **:include** alias list.

For details on setting up mailing lists, see the **list**(ADM) manual page.

The alias.user file

To map users to specific machines, specify aliases in the *alias.user* file. Use this file when you want each person at your site to receive mail on a particular machine. The following example shows how to set up *alias.user*:

```
andrei: andrei@scribe
george: george@dera
hanna:  hanna@scribe
karen:  karen@guardian
```

The alias.ali file

Use the *alias.ali* file for aliases that are not specific to *alias.list* or *alias.user*. This file usually contains aliases related to system administration, such as:

```
root:          david
mmdf:          david
postmaster:    david
uucp:          david
```

In this example, all mail addressed to *root*, *mmdf*, *postmaster*, and *uucp* is redirected to *david* on the local machine.

In general, do **not** create aliases for remote users (users not located at the local site) in the *alias.ali* file.

How MMDF uses alias files

When mail is addressed to *postmaster*, MMDF routes the mail by first searching the hashed alias table from *alias.ali* to expand the *postmaster* alias to the associated user name. For example, the *postmaster* entry in the *alias.ali* file:

```
postmaster:    david
```

Then, MMDF searches the *alias.user* file to find the local machine name associated with the user name. For example, the *alias.user* file contains an entry like this:

```
david:  david@golem
```

MMDF then uses this information when searching the various *.dom* files, which map the local machine name to a fully qualified host name and the *.chn* files, which map the fully qualified host name to information on how to route the message. The section "How MMDF routes mail" explains the process for searching configuration files further.

Domain files

The domain files are used to match a host name to its fully-qualified host name. Domain files serve two purposes: to convey information to MMDF about how machines are connected, and to specify special routing considerations for subdomains or top-level domains.

Domain files are named for the domain that they describe (except for *root.dom*, which contains domain information not named in other domain files); the filenames end in *.dom*. The general practice is to have a separate domain file for the domain in which the host machine resides.

The operating system distribution includes four domain files in the */usr/mmdf/table* directory. Table 19.5 lists these files.

Table 19-5 MMDF domain files

Domain	Domain file	Describes
local	local.dom	local machine
domain	domain.dom	machines in the local domain
uucp	uucp.dom	machines in UUCP domain
root	root.dom*	domains not listed in other domain files

You can create new domain files for each domain. For example, create a domain file for the *npr.COM* domain and name it *npr.dom*. You do not need to use the **.dom* naming scheme; however, this filename extension makes it easier to determine the purpose of the file.

Domain file format

Each domain file consists of two columns of information: the left column lists the host name and the right column lists the fully qualified name for that host. The domain names can be either upper- or lowercase. Use tabs, spaces, a colon, or a combination of these characters to separate the first column from the second.

The name of a domain file determines the domain names for which MMDF searches. For instance, the domain file for UUCP generally contains entries for names in the following form:

```
machine:         machine.UUCP
```

However, you can create an entry to map a specific UUCP address to another address. For example, to map *research.UUCP* to *research.jcn.com*, the entry looks like the following:

```
research:       research.jcn.com
```

The following four sections give examples of different domain files.

The local.dom file

The *local.dom* file describes the local host. For example, the contents of the *local.dom* file on the host *scribe* in the domain *npr.COM* look like the following:

```
scribe: scribe.npr.COM
```

or

```
scribe: npr.COM
```

if you selected to hide the host machine name.

* The *root.dom* file also contains information about how to access top-level domains, such as *MIL* and *GOV*. (The name *root* implies "top-level", as in a hierarchy).

The domain.dom file

The *domain.dom* file describes the machines in the local domain (independent of the channel that MMDF uses to reach each machine). For example, if there are four machines in the domain *npr.COM,* the *domain.dom* file looks like the following:

```
scribe: scribe.npr.COM
huey:   huey.npr.COM
dewey:  dewey.npr.COM
louie:  louie.npr.COM
```

Note that the *domain.dom* file maps each machine to the fully qualified host name in the *npr.COM* domain.

The uucp.dom file

The *uucp.dom* file specifies the hosts in the UUCP domain. For example, if your host connects to the remote machine *cactus* via UUCP, the *uucp.dom* file looks like the following:

```
cactus: cactus.UUCP
palm:   palm.UUCP
```

In this case, MMDF directs any mail sent to the *cactus* or *palm* machines to the UUCP network.

The root.dom file

The *root.dom* file defines the hosts and domains not defined in the other domain files. For example, if your host connects to the UUnet network system, you can set up MMDF to send all mail to specific domains to *uunet.UU.NET.* To do this, set up your *root.dom* file like the following:

```
COM:    uunet.UU.NET
EDU:    ucscc.EDU
MIL:    uunet.UU.NET
GOV:    star.GOV
NET:    uunet.UU.NET
```

In this example, all mail directed to the *COM, MIL* is sent out on UUnet; mail to *GOV* goes to *star.GOV;* and mail sent to the *EDU* domain goes to *ucscc.EDU.*

LAN considerations

If you are configuring MMDF for use on a local area network (LAN), you can use the domain files to distribute the processing load on the machines, or you can designate a special mail server machine to route all the messages.

You have the following choices:

- MMDF Server — Designate one machine as the network server. (This machine may also have outer world access.)

  Each machine's domain file only needs to include **badhosts** and **badusers** channels that contain the fully qualified host name of the server. The domain file can be the same for each machine in the network. The network can grow, and machines can be added and removed with no effect on the domain files of other computers. The disadvantage is that the server receives a great deal of traffic and should be dedicated to its task. In addition, if the server is down, so is all electronic mail between machines. The best policy is a system of machines grouped around a server that has knowledge of another server; each server has knowledge of yet another and so on.

- Distributed Processing — Give each machine's domain file knowledge of each other machine in the network.

  The advantage is that networked machines can operate independently of each other. One machine's crash has no effect on the mail capability of the others (unless that machine is connected to another network via UUCP). The disadvantages are that system administration gets geometrically more difficult as you add or remove machines to or from the network. When a machine or user is added to or removed from the network, you must update all domain or alias files to recognize the change. Because a domain file can contain redundant information about the local machine, you can use one domain file on every machine. You should only use distributed processing for small networks.

  You can greatly simplify the distributed processing configuration by running a name server program on your network. For details on setting up the name server, See Chapter 4, "Name Server Operations Guide for BIND," in your *TCP/IP Administrator's Guide*.

- Gateways — Use a gateway to connect a network to another Local Area or Wide Area network. (This setup is derived from the server setup.) In this case, the server machine is typically connected to more than one network.

  In addition to containing the names of the local machines on the LAN, a gateway machine also contains the names of the other machines to reach over the other networks. This information is kept in the respective domain and channel files for the other networks and also in the *root.dom* file on the gateway machine. Other machines on the LAN use the **badhosts** channel to route non-local mail to the gateway machine, or a *root.dom* file that lists all the top-level domains as routing through the gateway. To avoid overloading the gateway machine, the other machines on the LAN use local domain files as described in the earlier "Distributed Processing" bullet.

Channel files

MMDF uses the channel files to determine the channel to use for outgoing mail and the address of the host on that channel. Channel files map the fully qualified host name (as determined from the domain file entries) to channel-specific addressing information. For example, the UUCP channel file maps host names to UUCP paths (using exclamation points) specifying how to get to each host.

The operating system distribution includes four channel files in the */usr/mmdf/table* directory. Table 19.6 lists these files.

Table 19-6 MMDF channel files

Channel	Channel file	Describes
local	local.chn	local machine
list	list.chn	list-processor
smtp	smtp.chn	machines accessed via SMTP
uucp	uucp.chn	machines accessed via UUCP

The **MCHN** definitions in the */usr/mmdf/mmdftailor* file direct MMDF to search the specified *.chn* files in the */usr/mmdf/table* directory for channel definitions.

Separate the left and right columns in the channel files by a space or tab, a colon character " : ", or both.

The local.chn file

The *local.chn* file contains entries describing all the names local host is called, mapping them to the local host name. For example, if the local host is *scribe.npr.COM,* then the local host is known as *scribe* on the local machine. The *local.chn* file maps the local host name (on the right) to the different ways people might refer to *scribe.* This file looks like the following:

```
scribe:         scribe
npr.COM:        scribe
scribe.npr.COM: scribe
```

The list.chn file

The *list.chn* file contains information about the **list-processor** program:

```
list-processor: list-processor
list-proc:      list-processor
```

The left column is a pseudo-host defined in a mailing list alias (see "The alias.list file" section in this chapter). These entries tell MMDF to pass mail addressed to a mailing list to the **list-processor** program.

The uucp.chn file

The *uucp.chn* file contains entries describing the hosts that your host connects to using the UUCP channel and how to route mail to those hosts. For example, the format of this file looks like the following:

```
mcvax.UUCP:     uunet!mcvax!%s
sri-nic.ARPA:   uunet!sri-nic.arpa!%s
uunet.uu.NET:   uunet!%s
```

The left column contains the UUCP host name from the domain tables; the right column describes the UUCP address that MMDF uses to direct mail to that host. The "%s" at the end of the UUCP address means to use the rest of the address from this point on. In other words, when mail is addressed to the user *hillis* at *mcvax.UUCP*, the UUCP channel passes the mail to UUnet along with the rest of the UUCP address (*mcvax!hillis*). The second entry in this example shows how a domain name (*sri-nic.ARPA*) can be used within a UUCP path.

Channel file entries for the UUCP channel (in the *uucp.chn* file) when the destination machine is multiple hops away, appear as follows:

```
stooges.UUCP:   moe!curly!larry!stooges!%s
```

Specify the ***address*** of the host on the right-hand-side, where the ***address*** is a UUCP path.

The smtp.chn file

The *smtp.chn* file describes the hosts that you connect to using TCP/IP and the IP addresses of those hosts. The format of this file looks like the following:

```
cocoa.npr.COM:          123.456.789.1
caramel.npr.COM:        123.456.789.2
taffy.npr.COM:          123.456.789.3
```

The left column contains the fully-qualified host names for the hosts that you connect to with TCP/IP and the right column contains the IP addresses. For more information about configuring MMDF to route mail over TCP/IP, see your *TCP/IP Administrator's Guide.*

Following the pattern of mapping the host name in the left column to the addressing information for delivering to that host in the right column, you can create channel files for each **MCHN** definition in *mmdftailor*. The exception to this is the **badhosts** pseudo-channel definition; the **badhosts** channel program is determined at configuration time automatically. However, you can set up your channel files to indicate the channel that you use to reach the "smart host".

How MMDF routes mail

This section describes how MMDF uses the information in the configuration files to route mail on your system. Note that MMDF never searches the alias, channel, and domain files directly. When you build the hashed database with **dbmbuild**(ADM), the contents of these files are stored in tables in the *dbm* database. MMDF then uses the information in this database to route mail.

Mail arrives at and leaves computers using one of several different methods (such as UUCP, or TCP/IP called "channels". The MMDF **submit**(ADM) command accepts the incoming mail from a channel and determines the correct outgoing channel to use based on the destination host. The **submit** program uses the information in the domain tables to map the way the incoming mail describes the destination host to the way the host recognizes that destination host. Based on the host description, **submit** uses the channel tables to determine the outgoing channel to use to route the message and places the message in the appropriate queue.

Then, using the information from */usr/mmdf/mmdftailor*, **deliver**(ADM) moves the mail from the queue to the appropriate channel. For example, if **submit** places a message in the UUCP channel queue, **deliver** moves that message to the UUCP channel. The **deliver** command can also place mail in a channel and let another program (such as **uux**(C) for the UUCP channel) carry out additional steps to resolve the circumstances dictated by the type of channel. The channel program sends the mail across the network to the proper destination.

Searching MMDF domain tables

The **submit**(ADM) command uses the domain tables for two purposes: to specify the fully-qualified host name and (optionally) to specify the route to a host by listing the fully-qualified host names of one or more intermediate hosts through which mail is to be routed.

First, **submit** separates the fully-qualified host name into two parts: the name of the domain table and the hostname to match on the left-hand side (LHS) of the entries in the domain table. For example, in the address *david@engr.canada.COM*, the name of the domain table to search is *canada.COM* and the hostname to search for is *engr*.

MMDF tests an address for matches against the domain names in the **MDMN** entries in *mmdftailor*. For example, the address *david@engr.canada.COM* matches the following **MDMN** entry:

```
MDMN   "canada.com", show="Canada Delivery", tbl=canadadom
```

Then, MMDF searches *mmdftailor* for the **MTBL** entry for *canadadom*:

```
MTBL  canadadom,  file="canada.dom",   show "Canada Delivery"
```

MMDF uses the file associated with *canadadom* (*canada.dom*).

Thus, to route our message to *david@engr.canada.COM*, **submit** uses the following algorithm to search the domain tables for a match in the address:

1. Searches for the hostname (*engr*) in the LHS of the domain table.
2. Searches in the LHS of the domain table (in this case, *COM*), if it exists, for the domain name (*engr.canada*).
3. Searches the LHS of the *root* domain table for the fully-qualified host name (*eng.canada.COM*).
4. Searches the relevant tables that include ***flags=route***[2] in the **MTBL** line in *mmdftailor* for substrings of the address. For example, search the *COM* table for *canada*; search the *root* table for *canada.COM* or *COM*.
5. Searches the tables that include ***flags=partial***[3] in the **MTBL** line in *mmdftailor* for the input name, in this case, the fully qualified host name (*eng.canada.COM*), regardless of the domain name.

If **submit** finds no match at all, as a last resort, it uses the **badhosts** channel, if it exists. Because MMDF uses the first domain that has an exact match without looking for other matches in later tables, the order in which you list **MDMN** definitions is significant. Make sure the local domain table appears first and the *root* domain table is last in *mmdftailor*.

If the address provided is ***username@host*** (for example, *david@engr*), MMDF performs the same search as described above except that there is no *canada.COM* to match in the **MDMN** entries in *mmdftailor*. In this case, MMDF searches the *root* domain table and then the tables that include ***flags=partial*** in the **MDMN** line.

2. The ***flags=route*** parameter in the **MTBL** entry enables that table entry to match addresses for an entire subdomain, acting as a gateway. Do not use ***flags=route*** on tables other than the *root* domain table unless you have internal subdomains.

3. Use the ***flags=partial*** parameter on the *local* domain table so that users do not have to specify the full domain to send mail on the local machine.

Delivery channel programs

A channel is a compiled program that permits a machine to talk to a single type of network communications protocol. Some simple channels only store mail for further processing. Channel programs reside in the */usr/mmdf/chans* directory.

Channel programs handle the communications protocol so that neither the operating system nor the rest of the MMDF system has to know about the intricacies of a particular communication protocol. This handling of the protocol is one of the advantages of MMDF; having specific protocol program modules permits a site to upgrade to other network types without having to rewrite the mail delivery system.

The different channels are:

badhosts	Called when a specified machine is unknown to the local machine. The use of "bad" is really a misnomer; any mail to an unknown machine is sent on this channel. You can assign **badhosts** in the *mmdftailor* file to a channel type such as **uucp** or **smtp**.
badusers	Called when mail arrives at the local machine, but the user does not have a login on this machine, nor is there an alias for the user on the local machine. Generally, MMDF queues up this mail for submission to another machine with a larger list of users (a "smart host").
list	Called to remail messages. This channel simply invokes **submit** and feeds the addresses and text back into the MMDF mail system. This is often used to avoid long address validation or to force the validation to occur in the background for very large mailing lists. This also ensures that MMDF sends any problem reports to the list maintainer.
local	Called to deliver mail to mailboxes and processes on the local machine.
smtp	Called to deliver or accept mail from a TCP/IP network connection. The **smtp** channel transfers messages by establishing a TCP/IP connection to a remote machine, and using the Simple Mail Transfer Protocol (SMTP) to send one or more mail messages. The Internet Protocol (IP) allows many local- or wide-area networks to be interconnected transparently. This permits the MMDF SMTP channel to exchange messages with any machine on any network to which it is connected. For example, if your machine connects to the Internet, you can exchange messages directly with any machine in the world that is also connected to the Internet.

uucp Called to direct mail to UUCP delivery to another machine, or to accept mail from a UUCP connection from another machine. Incoming mail is converted into the format specified by the Internet technical bulletin, RFC822, available from the DDN Network Information Center, or NIC.* Outgoing mail includes a "From<space>" line and the mail path arguments are separated by UUCP exclamation point characters " ! ".

For more information about UUCP, see the "Building a remote network with UUCP" chapter of this guide.

Channels act not only as protocol handlers, but in some cases actually initiate the communications to the network or to another machine as needed.

For each channel, MMDF provides two programs: an input program and an output program. A channel program is associated with outgoing mail; a server program is associated with incoming mail. For example, the UUCP channel has */usr/bin/rmail* as its input program and */usr/mmdf/chans/uucp* as its output program.

On the MMDF system, the server is a channel program that monitors the network for incoming mail. On systems other than MMDF, the function of a mail server is built into the mail delivery system. Figure 19-3 describes the relationship between a channel and server.

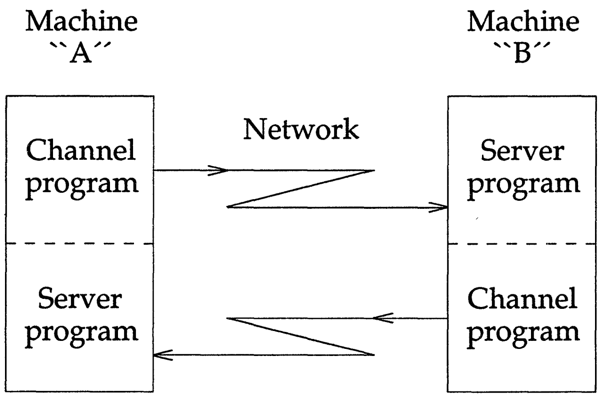


Figure 19-3 Channel and server relationship

* See the "Registering domain names" section earlier in this chapter for more information.

Even if MMDF is only being run on one side, the other side, while not correctly called a channel program, performs the same function.

Modifying MMDF configuration

When you perform the initial MMDF configuration, the configuration utility sets up the appropriate *alias.**, **.dom*, **.chn*, and *mmdftailor* files. This initial configuration is sufficient for most sites; if your site configuration changes, you can rerun the configuration utility to update the MMDF files.

However, if this initial configuration does not take care of your site configuration needs, you must edit the MMDF configuration files manually after running the configuration utility. You might also need to edit your configuration files manually if your site configuration changes so that you can no longer configure MMDF with the configuration utility. For more information, see the section "Configuration utility limitations" earlier in this chapter.

Guidelines for manual configuration

Keep in mind the following guidelines when you modify your MMDF configuration files:

- Always perform configuration as *mmdf*.
- Always rebuild the hashed database with **dbmbuild**(ADM) after modifying files. See the section "Rebuilding the MMDF hashed database" for more information.
- If you make any changes to the channel configuration (for example, to add or remove a channel), restart the **deliver**(ADM) daemon.

Changing the postmaster alias

Because RFC821/822 requires that every host provide the special postmaster mailbox, you should never change the **MSUPPORT** line in */usr/mmdf/mmdftailor* to anything other than *postmaster*. However, if the person responsible for mail administration at your site changes, you should change the *postmaster* alias in the */usr/mmdf/table/alias.ali* file.

For example, to change the *postmaster* alias to *hanna*, change the line in *alias.ali* to look like this:

```
postmaster:     hanna
```

NOTE You must define an alias (in the *alias.ali* file) that maps *postmaster* to a real user on the system.

You can designate anyone to receive undeliverable mail, but a local user is best because the address is simpler and, therefore, more likely to be a valid address. Also, with a local user, delivery of the message does not depend on a network connection that might be malfunctioning and therefore responsible for the original mail problem.

Maintaining user aliases

As you add new users to your system, you should update your */usr/mmdf/table/alias.user* file to map the new users to their home machines. For example, if you add *laurie* to the machine *poet* at your site, add the following line to *alias.user*:

```
laurie: laurie@poet
```

You can change the aliases in this file for other reasons, such as when you switch a user to another machine or the user leaves the company. For example, if a user leaves the company, you can redirect their mail to another person, such as their manager. To do this, place a line like the following in *alias.table*:

```
matt:  mark@scribe
```

If you create multiple-user aliases (mailing lists), you might need to update the information in your */usr/mmdf/table/alias.list* file periodically. For example, if you add a new alias called *docstyle*, add the following lines to *alias.list*:

```
docstyle:              docstyle-outbound@list-processor
docstyle-outbound:     joan,kelly,hanna,laurie,tammy,meg,teresa
docstyle-request:      joan
```

Changing the host name

When you install your operating system, the installation procedure prompts you to provide a name for your machine. This machine name is also called the HOST NAME or SYSTEM NAME.

If you decide to change your host name after installing the operating system, use the following procedure:

1. Log in as *root*.
2. Change the host name using the **uname**(C) utility. Use the following command, where *new_name* is the new name of your host:

   uname -S *new_name*

For example, to change a machine name to *elgrande,* enter the following command:

```
uname -S elgrande
```

For more information on the **uname** command, see the **uname**(C) manual page.

3. Now, log in as *mmdf* and verify that you are in the */usr/mmdf* directory. If not, change to the */usr/mmdf* directory with this command:

   ```
   cd /usr/mmdf
   ```

4. Edit the *mmdftailor* file and change all occurrences of the old host name to ***new_name.***
5. Change to the */usr/mmdf/table* directory and edit any **.dom* and **.chn* files, changing all instances of the old host name to ***new_name.***
6. Now, verify that you are in the */usr/mmdf/table* directory and rebuild the hashed database using the following command:

   ```
   ./dbmbuild
   ```

7. Now, log in as *root,* and bring the system down to single-user (system maintenance) mode with the following command:

   ```
   init 1
   ```

8. At the prompt, enter ⟨Ctrl⟩**d** to bring the system back up to multiuser mode.
9. If you are routing mail over TCP/IP, verify that the */etc/hosts* file contains the new host name.

Now, MMDF uses the new host name.

Hiding your host name

If, after running the initial MMDF configuration with the configuration utility, you register your top-level domain or subdomain with the NIC (Network Information Center), you might decide to hide the host name behind the domain name. In this case, people outside your organization can send mail to people at your site without knowing the host name where they receive their mail.

To hide your host name, you must edit the */usr/mmdf/mmdftailor* file and change the **MLDOMAIN, MLNAME,** and **MLOCMACHINE** keywords. For example, if you are not currently hiding the host name, these lines look like the following:

```
MLDOMAIN        npr.COM
MLNAME          scribe
```

To change the configuration to hide *scribe* behind the *npr* domain name, add **MLOCMACHINE** and change these parameters so they look like this:

```
MLDOMAIN  com
MLNAME npr
MLOCMACHINE scribe
```

Now, people can send mail to *andrei@scribe.npr.com* using the *andrei@npr.com* address (they no longer have to specify the host name, *scribe*).

Changing your smart host

If your configuration changes so that you no longer rely on a smart host for complete information about the network, or your smart host changes, you must modify the **MCHN** lines for **badhosts** and **badusers** in *mmdftailor.* These lines look something like the following:

```
MCHN badhosts, show="Last-Chance Routing", pgm=smtp, tbl=smtpchn,
     ap=822, host="rocky.npr.COM", confstr=scribe.npr.COM
MCHN badusers, show="Last-Chance Routing", pgm=smtp, tbl=smtpchn,
     ap=822, host="moose.npr.COM", confstr=scribe.npr.COM
```

In this case, change the value of the **host** parameter (in this example, *rocky.npr.COM* and *moose.npr.COM*) in each **MCHN** definition to the name of the new smart host. For more information about smart hosts, see the "Hide your host name?" section earlier in this chapter.

Changing error logging levels

By default, the MMDF configuration script sets logging levels with the **MMSGLOG** and **MCHANLOG** keywords in the *mmdftailor* file. These keywords are set to the FAT logging level (to log fatal errors only) and look like the following:

```
MMSGLOG         level=FAT
MCHANLOG        level=FAT
```

You can change these logging levels to a higher level, such as FST. In addition, you can use the **AUTHLOG** keyword to control authorization information. For a complete list of available logging levels, see the **MCHANLOG** section in *mmdftailor*(F).

Converting from XENIX

If you are converting from a XENIX to UNIX system or you have a mixed UNIX/XENIX system network, you must convert your XENIX aliases file and your UUCP routing files (as appropriate). (If you install your UUCP configuration files before running the MMDF configuration utility, you only have to convert the alias files.) This section explains how to convert these files.

Before converting your XENIX files, you must install and configure UUCP. You must install and configure these packages so that the files that you use to create MMDF alias, channel, and domain files contain the correct information.

If you did not install the entire distribution, you should install the UUCP and MAIL packages using the **custom**(ADM) utility. See your *Installation Guide* for information on installing packages with **custom.** For information on configuring UUCP, see the "Building a remote network with UUCP" chapter of this guide.

MMDF provides the utilities listed in Table 19.7 in the */usr/mmdf/table/tools* directory for converting files from XENIX to MMDF.

Table 19-7 MMDF Conversion utilities

Utility	Converts this XENIX file:	To these MMDF files:
mmdfalias	/usr/lib/mail/aliases	/usr/mmdf/table/alias.list /usr/mmdf/table/alias.user
uulist	/usr/lib/uucp/Systems	/usr/mmdf/table/uucp.chn /usr/mmdf/table/uucp.dom

For more information, see the **mmdfalias**(ADM) and **uulist**(ADM) manual pages. Note that **uulist** uses **uuname**(C) to retrieve the list of accessible sites.

Use the following procedure to convert the XENIX files:

1. First, use the following command to make sure *aliases* and *top* are readable by *mmdf*:

   ```
   ls -l /usr/lib/mail/aliases /usr/lib/mail/top
   ```

   If the *mmdf* user does not have permission to read these files, use **chmod**(C) to change the permissions. Log in as *root* and enter the following command:

   ```
   chmod +r /usr/lib/mail/aliases /usr/lib/mail/top
   ```

2. Now, log in as *mmdf* and enter the following command:

   ```
   cd /usr/mmdf/table
   ```

3. Use the following commands to convert the routing or alias files, as appropriate:

   - To convert alias files, enter:

     ```
     tools/mmdfalias
     ```

     The **mmdfalias** utility creates the files *alias.list* and *alias.user* in the */usr/mmdf/table* directory.

- To convert UUCP routing files, enter:

  tools/uulist

  The **uulist** utility creates the files *uucp.chn* and *uucp.dom* in the */usr/mmdf/table* directory.

If any of the MMDF output files already exist, the MMDF utility backs up the original *filename* to *filename-*.

4. Now, make sure that the permissions on each of the output files are set to 644, and that the owner and group are *mmdf*. Use the following commands:

   ```
   cd /usr/mmdf/table
   ls -l alias.list alias.user uucp.chn uucp.dom
   ```

   The permissions should look something like the following:

   ```
   -rw-r--r--   1 mmdf     mmdf         150 Jun 11 1990 alias.list
   -rw-r--r--   1 mmdf     mmdf         120 Jun 11 1990 alias.user
   -rw-r--r--   1 mmdf     mmdf          86 Apr 17 14:23 uucp.chn
   -rw-r--r--   1 mmdf     mmdf          61 Apr 17 14:23 uucp.dom
   ```

   If the permissions for any of these files are wrong, use **chmod**(C) to modify them.

Now, all the XENIX routing and alias files are converted to MMDF.

If you are converting to MMDF from XENIX or another UNIX system, the system mailboxes for the users on the system are in the XENIX-style (older UNIX system) format. XENIX format uses the "From<space>" lines to delimit between messages in mail folders; MMDF uses ⟨Ctrl⟩**a** characters. You should convert the system mailboxes to use the new MMDF format using the **cnvtmbox**(ADM) utility.

Use the following procedure to do this:

1. First, verify that users on the system are not accessing the system mailboxes.
2. Log in as *root*. (You must have read permission on the mailbox that you are converting.)
3. Now, change to the system mail spool directory (generally, */usr/spool/mail*).
4. For each system mailbox in the mail directory, run the following commands:

   ```
   /usr/mmdf/bin/cnvtmbox old_mailbox new_mailbox
   mv new_mailbox old_mailbox
   ```

By default, the **mailx**(C) program maintains the current format of every mail folder that users read with **mailx**. For example, if a mail folder is in MMDF format (with ⟨Ctrl⟩**a** delimiters), **mailx** uses this format when adding messages to the folder. If a user runs **mailx** with a folder that has not been converted to MMDF format, it prompts the user to convert the folder.

If any users on the system use an MUA other than **mailx**, advise these users to run **cnvtmbox** on each of their mail folders. As long as users have read permission on a mail folder and write permission on the output, they can convert it to the MMDF format with **cnvtmbox**.

Modifying UUCP host configuration

If you change your UUCP host configuration, use the information in this section to update your MMDF configuration files.

Specifying options to uux

If you use UUCP to route mail, you can specify the options that **uux**(C) should invoke when sending files and executing commands on the remote machine. By default, MMDF uses the following options to **uux**:

```
uux - -r
```

This setting specifies that UUCP queue the job until the next time **uucico**(ADM) runs rather than transfer it immediately (see **uucico**(ADM) for more information).

If you want to change this, add the **UUXSTR** keyword to the */usr/mmdf/mmdftailor* file. For example, if you want **uucico** to run immediately, the following **UUXSTR** setting is correct:

```
UUXSTR "uux -"
```

In general, we recommend that you use one of the following **UUXSTR** settings:

```
UUXSTR "uux -"
UUXSTR "uux - -r"
UUXSTR "uux - -r -gA"
```

For more information on these options, see the **uux**(C) manual page.

Adding or removing a UUCP host

To add a host that you access via UUCP to your system, use the following procedure:

1. Add the UUCP path to route mail to the host to the */usr/mmdf/table/uucp.chn* file. For example, to add the UUCP host *dudley*, add the following line to *uucp.chn* on the local host:

   ```
   dudley.UUCP: dudley!%s
   ```

   The %s specifies to use the remainder of the address indicated in the "To:" mail header from this point.

2. Add the UUCP host name to the *uucp.dom* file. For example:

   ```
   dudley:        dudley.UUCP
   ```

3. Now, rebuild the hashed database with **dbmbuild**.

To remove a UUCP host, remove the entries for that host from the *uucp.chn* and *uucp.dom* files and rebuild the hashed database.

Adding or removing an SMTP host

Use the following procedure to add an SMTP host to your MMDF configuration:

1. For each new host in */etc/hosts*, add the fully qualified host name and IP address to the */usr/mmdf/table/smtp.chn* file. For example, to add two new hosts *chocolate* and *truffle*, enter:

   ```
   chocolate.npr.COM:   123.456.789.4
   truffle.npr.COM:     123.456.789.5
   ```

2. Now, edit the */usr/mmdf/table/smtp.dom* and match the unqualified SMTP host name to the fully qualified host name. For example, enter:

   ```
   chocolate:     chocolate.npr.COM
   truffle:       truffle.npr.COM
   ```

3. Now, use **dbmbuild**(ADM) to rebuild the MMDF hashed database.

Use the information in the section "Testing MMDF configuration" later in this chapter to check the new SMTP configuration. For more information on routing mail over TCP/IP, see your *TCP/IP Administrator's Guide.*

Running multiple deliver daemons

If you configure MMDF to route mail on channels in addition to the local channel, you must restart the **deliver** daemon and modify the */etc/rc2.d/S86mmdf* system startup file to tell **deliver** about the new channels.[4] The **deliver** daemon periodically checks each channel's queue for mail to deliver. This is known as "sweeping the queues".

The "Modifying MMDF system startup" section explains how to start one **deliver** process for all the channels that you have configured. When you have a single **deliver** program managing a number of channels, **deliver** goes through the channels individually and tries to deliver all the messages in a channel's queue before going on to the next channel. You can configure your system to start multiple **deliver** programs, each servicing a single channel, instead. In this case, the **deliver** daemons work in parallel. This is a desirable configuration for a mail gateway machine because it increases the overall mail bandwidth of the machine.

NOTE For each **deliver** process that you initiate, system overhead increases.

To start separate **deliver** processes to manage each channel, use the following information to add **deliver** startup lines for each channel to the */etc/rc2.d/S86mmdf* file.

If you removed the **-clocal** from the **deliver** startup line in your *S86mmdf* file, the **deliver** startup line looks like the following:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b"
```

To create separate **deliver** processes, use the **-c** option to add lines for each channel. The following example shows separate **deliver** processes for each of the local, UUCP, and SMTP channels:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -clocal"
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -cuucp"
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -csmtp"
```

4. See the section "Modifying MMDF system startup" earlier in this chapter for information on how to do this.

If you have a lot of mail traffic on a single channel, you can start more than one **deliver** process running on that channel. To do this, simply add more than one **deliver** startup line for the channel in */etc/rc2.d/S86mmdf*. For example, to start three **deliver** processes to sweep the queue for the SMTP channel, add these lines:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -csmtp"
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -csmtp"
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -csmtp"
```

Keep in mind that each **deliver** process that you start increases the load on the system.

Specifying different time intervals for deliver

If you create a separate **deliver** daemon for each channel, you can set up a different time interval for each **deliver** process. To do this, add the **-T** option to the appropriate **deliver** startup line in */etc/rc2.d/S86mmdf*. For example, to run **deliver** on the local channel every 30 seconds and on the UUCP channel every 5 minutes, put the following lines in *S86mmdf*:

```
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -clocal -T30"
/bin/su mmdf -c "/usr/mmdf/bin/deliver -b -cuucp -T300"
```

Rebuilding the MMDF hashed database

The hashed database gives MMDF quick access to alias and routing information. You must rebuild this database whenever you change *mmdftailor* or any of the alias, channel or domain tables.

To rebuild the database, log in as *mmdf* and run the following commands:

```
cd /usr/mmdf/table
./dbmbuild
```

If any of the files in */usr/mmdf/table* are missing when you rebuild the database, **dbmbuild** reports that they are missing. See the **dbmbuild**(ADM) manual page for full details.

Testing MMDF configuration

Once you've modified your MMDF configuration and rebuilt the hashed database, use the information in this section to test the new configuration.

Checking for MMDF problems

Use the **checkup**(ADM) command to examine the full MMDF system on the local machine and report any inconsistencies. Normally, **checkup** reports all problems, including correct states; **checkup** marks problems with two asterisk characters "**" and and encloses advisory information in square brackets.

You can tell **checkup** to report only the problems using the **-p** option. For example, enter the following command:

```
/usr/mmdf/bin/checkup | more
```

The following example shows the output from **checkup**:

```
**  Asterisks indicate potentially serious anomalies.

Tailor file            : /usr/mmdf/mmdftailor
**    Unknown tailor   : 'MLCKTYPE advisor'
```

(In this case, **checkup** does not recognize the **MLCKTYPE** parameter **advisor** because it is misspelled (the correct parameter name is **advisory**).

If the permissions on any of the MMDF configuration files are incorrect, **checkup** lists this information.

Testing mail addresses

Use the **checkaddr**(ADM) command to test an individual address for validity. For example, to test the validity of the address, *andrei@scribe,* enter the following command:

```
/usr/mmdf/bin/checkaddr andrei@scribe
```

The **checkaddr** checks the address and displays the following message:

```
andrei@scribe: OK
```

This command is particularly useful for testing aliases after you have installed a new alias file. For more information, see the **checkaddr**(ADM) manual page.

Maintaining the MMDF system

Once you configure MMDF, the MMDF system requires minimal system administration. In addition to maintaining the alias files (as described in the "Maintaining user aliases" section earlier in this chapter) and adding new machines to your network (see the "Modifying MMDF configuration" section), you must perform the following tasks:

- Monitor space in the */usr/spool/mail* directory where mail is received, and in the */usr/spool/mmdf* directory where queued mail is stored.
- Clean outdated mail from the mail queues.
- Remove log files periodically from the */usr/mmdf/log* directory.

The following sections describe the utilities you can use to maintain your MMDF system.

Checking the status of mail queues

Use the **checkque**(ADM) program to check the status of the mail queues; this utility reports on the number of messages waiting for delivery. For example, enter the following command:

/usr/mmdf/bin/checkque

Here is an example of output from **checkque**:

```
Thu May  9 16:07:  2 queued msgs / 80 byte queue directory
                   4 Kbytes in msg dir


  2 msgs    4 Kb (local   ) local       :  Local Delivery
                 deliver start          :  Thu May  9 15:15
                 deliver message        :  Thu May  9 15:15
                 deliver end            :  Thu May  9 15:15 / 0 hours
  *** WAITING **  First message         :  Wed Apr 24 16:03
```

If a queue is backed up with waiting mail, you can manually force **deliver**(ADM) to deliver the mail using a command like the following:

deliver -w -clist,uucp

The **-c** option specifies the channels to be processed and **-w** forces **deliver** and the channel programs to output informative messages as they try to deliver the mail. You can review the output for abnormalities, such as a rejected sender or recipient. For a complete description of **checkque**, see the **checkque**(ADM) manual page.

Removing old mail from the queues

Use the **cleanque**(ADM) program to remove outdated files from the mail queues. Use **crontab**(C) as *mmdf* to run **cleanque** daily. You might want to run **cleanque** more often, depending on your mail volume. You can also run **cleanque** using **cron**(C). To do this, create a *crontabs* file for *mmdf* in the */usr/spool/cron/crontabs* directory that includes the following line:

```
0    0    *    *    0-6    /usr/mmdf/bin/cleanque
```

You can also run **cleanque** by hand whenever you suspect a problem with mail delivery. The **cleanque** manual page provides a complete description of this program.

Monitoring the size of log files

You should periodically check the size of log files in */usr/mmdf/log*. To limit the amount of log data that accumulates in a log file, move it to another filename, for example *chan.log-*.

If you move aside a log file, make sure that *mmdf* owns the new empty file. To back up the log file and create a new empty file, enter the following commands as *mmdf*:

```
cd /usr/mmdf/log
cp chan.log chan.log-
cat /dev/null > chan.log
```

Advanced MMDF configuration tasks

This section describes some additional configuration tasks that you can use to customize your MMDF system. In general, most sites do not need to perform any of the configuration procedures described in this section.

Changing the location of system mailboxes

By default, MMDF delivers mail to a file named with the user's login in the */usr/spool/mail* directory. If you want MMDF to deliver mail to a file or directory other than the default file, you can add lines (anywhere in the *mmdftailor* file) using the following **MDLVRDIR**, **MMBXNAME**, and **MMBXPROT** keywords.

Only one of **MDLVRDIR** and **MMBXNAME** can be non-null. If **MDLVRDIR** is null, MMDF delivers to the user's home directory; if **MMBXNAME** is null, MMDF uses the user's login as the name of the mailbox file. You can also change the location of the spool directory by setting **MDLVRDIR** to be non-null.

MMBXPROT sets the protection mode on mailbox files using the same set of octal numbers that you use to change access permissions on files with **chmod**(C). For example:

```
MDLVRDIR ""
MMBXNAME ".mailbox"
MMBXPROT 0600
```

In this example, MMDF delivers to a *.mailbox* file in the user's home directory and sets the file protection so that only the owner can read or write to the file.

Specifying the MMDF "signature"

You can change the signature that MMDF uses when notifying senders of mail delivery problems by setting the **MSIG** keyword in the */usr/mmdf/mmdftailor* file. This message should indicate which mail routing system was responsible. For example:

```
MSIG     "MMDF Mail System"
```

With this setting, when users receive "Failed" mail messages from MMDF, the message appears to be from "MMDF Mail System".

Configuring MMDF authorization

The MMDF authorization feature allows you to control the flow of mail through your host. For example, if your site is on the Internet (you registered your domain name with SRI), you might want to allow your employees (but no one else) to send mail from their home machines through your system to the Internet. In this case, you would want to set up MMDF to authorize mail from those users to pass through your machine, but prevent mail from other users from passing through.

With MMDF, you can control mail transferred through your host on a per-channel or per-user basis (or both). In other words, you can control the flow of mail on specific channels and from specific users.

To control authorization on a particular host, use the **auth** parameter to set an authorization level on the **MCHN** line in the */usr/mmdf/mmdftailor* file. MMDF provides the seven levels of authorization listed in Table 19.8.

Table 19-8 MMDF authorization levels

Level	Description
free	performs no authorization checks (default)
inlog	logs incoming authorized and unauthorized access, but allows mail to pass
outlog	logs outgoing authorized and unauthorized access, but allows mail to pass
inwarn	logs incoming unauthorized access, transfers the message, and sends a warning to the originator
outwarn	logs outgoing unauthorized access, transfers the message, and sends a warning to the originator
inblock	logs the incoming unauthorized access attempt and bounces the mail
outblock	logs the outgoing unauthorized access attempt and bounces the mail

The default channel authorization level is **free**; any channel that you do not set authorization for is set to **free**.

The following sections explain more about using these authorization levels to set up host- and user-based authorization.

Setting authorization for hosts

You can control the flow of mail both to and from your host and network by setting up host-based authorization controls. The most common reason that sites use host-based authorization is to make sure that hosts which are not part of a private network do not use that network to send mail to other parts of the world. For example, you can use host-based authorization to allow mail to enter your domain, but restrict the mail passing through it.

By default, MMDF performs all authorization on the next "hop" host in the address; if someone specifies a route through your host, MMDF on your host can authorize the next hop in that route. In other words, MMDF does not perform authorization based on the final destination unless the final destination is the next hop. However, you can perform authorization on the entire route to the destination or from the source using the **dho** flag with the **auth** parameter to specify routing-based authorization controls. See the "Routing-based authorization" section later in this chapter for more information.

You can control access to a particular channel based on whether the mail is inbound or outbound on the channel and whether the source or the destination of the mail is authorized. To do this, use one (or more) of the four tables listed in Table 19.9.

Table 19-9 Host-based authorization tables

Table	Description
insrc	when mail arrives from a particular host (the source host), this table verifies that that host is allowed to send mail to the destination host by authorizing either the destination host or the channels used to access the destination host
outsrc	when mail arrives from the source host via a particular channel, this table determines if that source host or channel is allowed to send mail to the destination host
indest	when mail is sent to a particular host or via a particular channel, this table determines whether the source host is allowed to send mail to the destination host or via that channel
outdest	when mail is sent to a particular host, this table verifies that the source host or channel is allowed to send mail to the destination host

For example, you might want to set up MMDF so that only hosts that belong to a certain network can send mail out on a specific channel. This could be the case if your site has limited resources and you do not want to allow other sites to pass mail through your site.

Use the following procedure to set up the host-based authorization controls:

1. First, log in as *mmdf*. (You must perform all MMDF configuration as *mmdf*.)
2. Use the **MTBL** keyword to define a table in */usr/mmdf/mmdftailor*. For example:

   MTBL "world-auth", file="authinfo/world"

   The *authinfo/world* table specifies the privileges for all the hosts that do not belong to the network. This table is stored in the file *world* in the directory */usr/mmdf/table/authinfo*; you can specify locations like this for all the tables.
3. Now, use MCHN to define two channels in *mmdftailor*: one channel for the hosts that belong to the network, one for the hosts that do not belong.

   For example, define a channel for the *sconet* network:

   MCHN sconet, auth=free, show="SCONET Delivery",
   ap=822, mod=imm

   (The "auth=free" authorization setting is the default; you do not have to specify it explicitly as in this example.)

   Now, define a channel for all the hosts (**world**) not in the *sconet* network (this appears as one line in *mmdftailor*):

   MCHN world, auth=inblock, indest="world-auth",
   show="WORLD Delivery", ap=822, mod=imm

   In this case, anyone can send mail out on the **world** channel, but MMDF checks the *world-auth* table to authorize the destination of mail arriving on this channel.
4. Now, create the channel tables (**.chn* files in the */usr/mmdf/table* directory) for the channels that you define in *mmdftailor* and include descriptions of each host accessed via that channel. (See the "Channel files" section in this chapter for more information.)
5. Finally, create the *world-auth* table and include lines like the following:

   ```
   world:
   local:
   sconet: moocow.uucp
   ```

   The entries on the LHS of the authorization table specify that if the destination for a message is on either the **world** or **local** channels, MMDF authorizes anyone using the **world** channel as an input channel to send mail.

   The entry on the RHS of the **sconet** channel entry specifies the hosts and channels that are authorized to send outgoing mail using the **sconet** channel. In this case, *moocow.uucp* is the only machine allowed to pass mail into the *sconet* network.

6. Now, rebuild the hashed database using the information in the section "Rebuilding the MMDF hashed database."

Routing-based authorization

You might want to set up authorization to keep mail that enters your domain from leaving your domain. In other words, people can send mail from inside the domain to people outside the domain as well as to other people in the domain, but people outside the domain cannot send mail through your domain to another destination.

For example, if your domain includes machines in different cities with links to different outside computers, people from outside the domain might use these links to send mail from one city to another, through your domain. In this case, you can set up MMDF authorization to prevent people from using your system to transfer mail.

Because MMDF performs authorization only on the next "hop" in the address, you cannot use the procedure described in the previous section to set up authorization control. To do this, you must use the **auth** parameter with the **dho** flag to specify routing-based authorization controls.

The following procedure explains how to set up routing-based authorization for the hosts that are not in the *nprnet* domain:

1. Log in as *mmdf*.
2. Define a table in */usr/mmdf/mmdftailor*. For example:

   ```
   MTBL "world-auth", file="authinfo/world"
   ```

   This defines the *authinfo/world* file that contains the authorization information for the **world** channel.
3. Now, specify a channel for your domain. For example, if the domain name is *npr.com*, create an **MCHN** entry like the following:

   ```
   MCHN nprnet, auth=free, show="NPRNET Delivery",
            ap=822, mod=imm
   ```

4. Now, define a channel for the rest of the hosts that are not in the local domain (again, this appears as one line in *mmdftailor*):

   ```
   MCHN world, auth=inblock, auth=dho, indest="world-auth",
            show="WORLD Delivery", ap=822, mod=imm
   ```

   The "auth=indest" parameter specifies that when **world** is the input channel, MMDF checks the *authinfo/world* file to verify that the inbound host is authorized to send mail to the destination.

When you specify the "auth=dho" parameter on a channel, MMDF replaces the "host" (in host-based authorization) used to check authorization with a route. The route is either from the source or to the destination, depending on which flag (from Table 19.9) that you specify. MMDF replaces the local section of the route (the user's name) with the string "username". Then, MMDF compares this route to the route specified in the message, to determine if the message is authorized or not.

5. Create the channel tables in the */usr/mmdf/table* directory for the channels that you define in *mmdftailor*. (See the "Channel files" section in this chapter for more information.)
6. Finally, create the *authinfo/world*, and include entries like the following:

```
world:
username@npr.com:
username@larry.npr.com:
username@moe.npr.com:
username@curly.npr.com:
```

   This table authorizes MMDF to deliver any mail addressed to people in the *npr.com* domain arriving or leaving on the **world** channel. This does not allow mail to pass through the **nprnet** channel to a destination outside the *npr.com* domain.

7. Now, rebuild the hashed database with **dbmbuild**.

Setting authorization for users

To configure MMDF authorization on a per-user basis, you must first set up the authorization level on the channels that you want to restrict as described in the previous section. Once you set up channel authorization levels, you must create a table in the */usr/mmdf/table* directory that maps user names to authorization levels and then declare this table in the */usr/mmdf/mmdftailor* file.

The following is the format for setting up the per-user authorization table:

***username*: *keyword channel* [, *channel*]**

The ***username*** can be either a local or remote user name, ***keyword*** is one of the keywords for the actions that you can authorize users to perform, and ***channel*** is the channel name from the **MCHN** line.

Table 19.10 lists the action authorization keywords:

Table 19-10 MMDF authorization keywords

Keyword	Description
both	allows user to send and receive mail
send	allows user to send mail only
recv	allows user to receive mail only
expire	expires access privileges for the user (and includes this information in any error mail)

MMDF treats any action not listed in the table above as **expire**, except that MMDF sends the text of the action instead of "expire" to the user in error mail.

Use these steps to set up authorizations for specific users:

1. First, log in as *mmdf*.
2. Use the information in the section "Setting authorization for hosts" to set up authorization for any channels that you want to restrict.
3. Declare the user authorization table in the */usr/mmdf/mmdftailor* file. For example, if the name of the user authorization table is *auth.user*, the **MTBL** declaration looks like this:

   ```
   MTBL auth, file="auth.user", show "Per-user authorization"
   ```

   Note that you must call the per-user authorization table "auth"; MMDF treats any table called "auth" as the per-user authorization table.
4. Now, create the user authorization table in the */usr/mmdf/table* directory. Edit the file and include one line for each user to whom you want to grant mail access. Any users not listed in the user authorization table are not authorized to use any channel except the ones set to the **free**, **inlog**, or **outlog** authorization levels.

   In our example, MMDF allows users who are not listed in the user authorization table to pass mail in and out through the UUCP channel and in through the TCP/IP channel.
5. To set up access authorizations for a local user, specify the unqualified user name. In the following example, the local user *andrei* can both send and receive mail on the TCP/IP and UUCP channels:

   ```
   andrei: both smtp,uucp
   ```

   However, if you set up host-based authorization to restrict access to one of these channels, for example UUCP, *andrei* might not be authorized to send or receive mail on that channel.

   Because mail on the local channel is not restricted, *andrei* can pass mail through this channel even though the user authorization list does not include "local" in the list of channels.

6. To set up access authorization for a remote user, specify the fully-qualified address of that user. For example, to allow *natasha* on the machine *kgb.GOV* to send mail through TCP/IP on this host, add a line like the following to the authorization table:

   ```
   natasha@kgb.GOV: send smtp
   ```

   If mail arrives for *natasha* through UUCP, or if she tries to send mail through the UUCP channel, MMDF rejects the mail.

   If you want to expire a particular user's access and tell MMDF to send an error message, add the message to the user authorization table line for that user. For example, to expire *rocky@squirrel.COM*'s access and send the text "No more mail for you!", include a line like this one:

   ```
   rocky@squirrel.COM: "No more mail for you!" uucp
   ```

7. Rebuild the hashed database with **dbmbuild**.

Setting host and user authorization controls

The default authorization algorithm allows MMDF to deliver a message if either the user-based or host-based authorization tables allow it. Thus, if a user table authorizes a user to send mail over a channel, but the host table does not, that user can still use the channel (and vice versa).

If you want to require that both the user and host tables authorize access to a channel, use the **hau** keyword with the **auth** parameter in the **MCHN** channel declaration in *mmdftailor*.

Understanding authorization log files

If you set the **level** parameter to the **AUTHLOG** keyword to an authorization level of FST (to give full statistics) or higher[5], MMDF maintains a complete log of authorization attempts and reasons for failure or success.

For example, to set the authorization logging level to save all diagnostic messages to the */usr/mmdf/log/mmdfauth.log* file, set **AUTHLOG** in the *mmdftailor* file like this:

AUTHLOG /usr/mmdf/log/mmdfauth.log level=FTR

5. See the section on MCHANLOG in the **mmdftailor**(F) manual page for a complete list of logging levels.

The format of authorization messages in the log file is similar to other MMDF log files. Each message includes the date, time, message source, and message ID, followed by the log-specific information. For example:[6]

```
4/29  9:44:54 AU-0000:  msg.a000561: i='local' o='ucscc'
      a='lisa@rsre.AC.UK' r='CH' hi='' ho='username@rsre.ac.uk'
4/29  9:44:54 AU-0000:  msg.a000561: i='local' o='ucscc'
      a='jane@rsre.AC.UK' r='CH' hi='' ho='username@rsre.ac.uk'
4/29  9:44:55 AU-0000:  msg.a000561: END size='2102', sender='robert'
```

Table 19.11 describes the authorization message keys.

Table 19-11 Authorization keys

Key	Description
i	input channel
o	output channel
a	destination address
r	reason for authorization
hi	inbound host
ho	outbound host

In addition, the authorization message includes an "end of processing" message that describes the message sender and size. (In the example above, the third line is the "end of processing" message.)

Each authorization message can have either one or two reasons for authorizing a particular message. If the authorization line includes one reason, the "r" key specifies a single authorization code which describes both the inbound and the outbound authorization when you use host-based authorization. These codes are listed Table 19.12.

Table 19-12 Single-reason authorization codes

Code	Description
OH	outbound host/route
HC	outbound host/route and inbound channel
HH	inbound host/route and outbound host/route
CC	inbound channel and outbound channel
CH	inbound channel and outbound host/route
IH	inbound host/route

6. The authorization messages appear on one line; the examples in this section split the lines for readability.

The following example uses the **CH** authorization code:

```
4/29  9:44:54 AU-0000:  msg.a000561: i='local' o='peaks'
      a='bob@rsre.AC.UK' r='CH' hi='' ho='username@rsre.ac.uk'
4/29  9:44:54 AU-0000:  msg.a000561: i='local' o='peaks'
      a='mike@rsre.AC.UK' r='CH' hi='' ho='username@rsre.ac.uk'
4/29  9:44:55 AU-0000:  msg.a000561: END size='2102', sender='cooper'
```

In this example, the authorized message has two recipients (*bob* and *mike*). The first authorization message shows that the inbound channel ("i") is the local channel and the outbound channel is **peaks**. The " a " key indicates that the recipient's address is *bob@rsre.AC.UK*.

The reason (" r ") given for authorizing the message is **CH**; in other words, the inbound channel (**local**) has authorization to send mail to the given outbound host or route (specified by " ho "), in this case *username@rsre.ac.uk*.

The two-reason authorization codes describe the reason for authorization in terms of user-based authorization. Table 19.13 lists these codes.

Table 19-13 Two-reason authorization codes

Code	Description
IL	inbound channel, outbound = **LIST**
OL	outbound channel, dest = **LIST**
IS	inbound channel by sender
OS	outbound channel by sender
IR	inbound channel by receiver
OR	outbound channel by receiver
I	inbound channel, log unauthorized access*
O	outbound channel, log unauthorized access*

The authorization message in the following example uses two-reason authorization (if no authorization is required for a channel, MMDF leaves the reason field (" r ") empty):

```
4/29  9:53:09 AU-0000:  msg.a000653: i='local' o='npr'
      a='john@edxa.ac.uk' r='' r='OS'
4/29  9:53:10 AU-0000:  msg.a000653: END size='197', sender='david'
```

In this example, the message arrived (with no authorization required) on the local channel and is authorized to leave on the **npr** channel because the sender (*david*) is authorized to use it (**OS**).

* MMDF only uses these authorization codes when you set "auth=inlog" or "auth=outlog".

Adding a new alias file

It is much easier to add entries to an existing alias file than to create one; see the sections "Alias files" and "Maintaining user aliases" earlier in this chapter for more information. If you want to add a new alias file, use the following procedure:

1. Log in as *mmdf.*
2. Create a file in the */usr/mmdf/table* directory.
3. Verify that the new alias file is owned and group-owned by *mmdf* and set the permissions to 644.

   NOTE Do not give write permissions on alias files to other users. With write permissions to alias files, a user can forward another user's mail to another location. For this reason, you should designate a trusted user to make changes to the alias files as requested.

4. Add alias information to the new alias file. See the "Alias files" section earlier in this chapter for information on the format of the various alias files.
5. Now, define the alias file in the */usr/mmdf/mmdftailor* file. Create an **MTBL** line to associate an abbreviated name and a more descriptive name with the new alias file. For example, the **MTBL** line in *mmdftailor* defines the alias file *alias.user* as:

   ```
   MTBL auser, file="alias.user", show="User Aliases"
   ```

   Now, you can refer to the file */usr/mmdf/table/alias.user* as "auser" throughout the rest of *mmdftailor.*
6. Add an **ALIAS** entry to *mmdftailor* to define information about the new alias file. For example, here is the **ALIAS** entry for *alias.user*:

   ```
   ALIAS table=auser
   ```

   The default parameters in the *mmdftailor* file are adequate for most requirements; however you can specify characteristics for the alias file using the following parameters:

   nobypass specifies that MMDF search the alias files first for all mail addresses on outgoing or incoming mail. In MMDF, placing a tilde " ~ " before an address causes the address to be interpreted as destined for its literal meaning. The **nobypass** parameter means that normal aliasing cannot be bypassed. For example, suppose the alias file contains this entry for George and **nobypass** is not set:

   ```
   george: george@vegan.edu
   ```

If you enter:

mail george

then the mail goes to the *george* login on the *astoria* machine. If you then enter:

mail "~george"

then aliasing is disabled and mail is sent to a George on the local machine. If nobypass is set, entering the tilde character " ~ " has no effect and every mail request is translated by the alias files.

public allows remote sites to expand aliases from this table using the SMTP **EXPN** command; you can also use **malias**(C) to display this information.

trusted directs mail to be delivered to any file or process using the permissions of any user on the system (including *root*); only the super user should have access to modify a trusted alias file.

Note that MMDF searches the alias tables in the order that you list them in *mmdftailor*.

For more information, see the **mmdftailor**(F) manual page.

Adding a new domain file

Use the following procedure to create a new domain file:

1. Log in as *mmdf*.
2. Create a file in the */usr/mmdf/table* directory. Generally, you should name domain files for the domain that they describe. For example, the domain file for the *npr.COM* domain is called *npr.dom*.
3. Verify that the new domain file has the correct permissions (see step 3 in the previous section).
4. Add the information for the domain to the new domain file. See the "Domain file format" section earlier in this chapter for more information.

   NOTE The entries on the left-hand side in this file must contain no leading whitespace.

5. Create an **MTBL** entry in *mmdftailor* for the new domain file. For example:

   ```
   MTBL nprdom, file="npr.dom", show="NPR Domain"
   ```

   See the **mmdftailor**(F) manual page for more information about additional parameters that you can use to specify additional properties about the table that you are defining with **MTBL**.

6. Add an **MDMN** entry for the new file:

   ```
   MDMN "npr.COM", show="NPR Domain", table=nprdom
   ```

   The first argument is the name of the domain.

Adding a third-party channel program

If you add a new channel program from an SCO add-on package, you must define and add a new channel file. A channel is the mechanism for delivering mail either to a mailbox on the local machine or across the network to a remote machine. The **MCHN** entries in *mmdftailor* define the channels available to MMDF for mail transport.

You can define additional channels for the network protocols configured on your system by specifying them in the *mmdftailor* file. Channel definitions look like this:

```
MCHN   uucp, show="UUCP Delivery", que=uucp,
       tbl=uuchn, ap=822, pgm=uucp, mod=imm
```

The first argument on the **MCHN** line is the name of the channel.

Use these steps to add a new channel:

1. Create an **MCHN** entry in */usr/mmdf/mmdftailor* and specify the parameters to use in your channel definition. For a complete list of the **MCHN** parameters, see the **mmdftailor**(F) manual page.
2. Create a channel file (**.chn*) in */usr/mmdf/table* and include entries for each host that you access via the new channel and how to route mail to those hosts. For more information, see the section "Channel files" earlier in this chapter.

Changing MMDF parameters

The MMDF parameters in the */usr/mmdf/mmdftailor* file allow you to redefine the certain MMDF variables. This section covers some of the more common MMDF parameters; for a complete list, see the **mmdftailor**(F) manual page.

AUTHLOG controls the authorization information. The following example shows the format of this parameter:

```
AUTHLOG /tmp/mmdf/mmdfauth.log, level=FST, size=40, stat=some
```

The **AUTHLOG** level must be at least FST, or MMDF does not save any authorization logging information. For more information, see the section on **MCHANLOG** in the **mmdftailor**(F) manual page.

MFAILTIME is the time (in hours) a message can remain in a queue before MMDF sends a failed mail message to the sender and purges the message from the queue.

```
MFAILTIME       168
```

MLCKTYPE allows you to specify the locking protocol for MMDF to use when locking users' mailboxes. This is useful if the users on the system use third-party MUA's that use a lock file that is different from the standard System V lock file. Set the **MLCKTYPE** parameter to one or more of the keywords in Table 19.14.

Table 19-14 MMDF locking keywords

Keyword	Lock file
advisory	System V fcntl() kernel locking protocols
v7	Version 7 and System V Release 3, and earlier locking protocols
xenix	XENIX locking protocols
all	all above locking protocols (default)

The default locking protocol is **all**; however the MMDF configuration utility sets **MLCKTYPE** to **advisory**.

With the **v7** keyword, MMDF creates a file called ***username****.lock* in */usr/spool/mail*. The **xenix** keyword specifies that MMDF create a file called */tmp/****username****.mlk*. In both cases, ***username*** is the name of the user's mailbox.

You can specify more than one locking protocol on the **MLCKTYPE** line. If you do, MMDF must successfully lock the user's mailbox using all the locking protocols before the mailbox is considered locked. For example, to use **advisory** and **xenix**, set **MLCKTYPE** like the following:

```
MLCKTYPE advisory, xenix
```

In this case, MMDF must lock the mailbox, using the **fcntl()** kernel file locking protocol *and* create a file called */usr/spool/mail/****username****.lock*. If it fails to perform both of these locks, MMDF must release the successful lock and try again later. Thus, if MMDF failed to perform the **fcntl** lock, it must unlink */usr/spool/mail/****username****.lock*.

MMAXHOPS specifies the maximum number of "Received:" or "Via:" lines a message can contain before the MMDF considers that the message is looping and rejects it.

```
MMAXHOPS        20
```

MWARNTIME specifies the time (in hours) that a message can remain in a queue before MMDF sends a warning message about delayed delivery to the sender.

```
MWARNTIME       72
```

MMSGLOG controls the logging information produced by the deliver **deliver**(ADM) and **submit**(ADM) programs. The following example shows the format of this parameter:

```
MMSGLOG /tmp/mmdf/mmdfmsg.log, level=FST, size=40, stat=some
```

For more information, see the **logs**(F) manual page and the section on **MCHANLOG** in the **mmdftailor**(F) manual page.

MCHANLOG controls MMDF logging, except for information controlled by **AUTHLOG** and **MMSGLOG**. See the **mmdftailor**(F) manual page for details.

Getting more information

If you need more information to understand how the various components of MMDF interact, refer to the manual pages listed in Table 19.15.

Table 19-15 MMDF manual pages

Manual Page	Description
checkaddr(ADM)	checks a mail address for validity
checkque(ADM)	reports on MMDF queue status
checkup(ADM)	checks MMDF system configuration
cleanque(ADM)	sends warnings and returns expired mail
cnvtmbox(ADM)	converts XENIX-style mailboxes to MMDF format
dbmbuild(ADM)	builds the MMDF database
dbmedit(ADM)	edits the MMDF database
deliver(ADM)	delivers mail to MMDF channels
list(ADM)	handles mailing lists
mmdf(ADM)	routes mail locally or over network
mmdfalias(ADM)	converts XENIX-style aliases file to MMDF
rmail(ADM)	submits remote mail received via MMDF
submit(ADM)	accepts mail to be handled by MMDF
uucico(ADM)	transfers UUCP files
uulist(ADM)	converts UUCP file to MMDF format
checkmail(C)	checks for submitted but not delivered mail
mail(C)	mails messages between users
uucp(C)	sends information between UNIX machines over serial lines
rcvalert(C)	alerts user when mail is received
rcvfile(C)	stores mail in a file automatically
rcvprint(C)	prints received mail automatically
rcvtrip(C)	informs sender that user is away
uname(C)	prints the name of the current system
uux(C)	transfers UUCP files and executes commands on remote system
logs(F)	logs MMDF messages
maildelivery(F)	describes the user delivery specification file
mmdftailor(F)	provides run-time tailoring
queue(F)	queues files for storing mail in transit
systems(F)	UUCP Systems file format
tables(F)	describes alias, domain, and host tables
llog(S)	performs standardized information logging for programs
ml_send(S)	provides simple interface for mail submission
mmdf(S)	simplifies MMDF MUA mail submission and pickup
phs(S)	notes the MMDF transmission phase
tai(S)	gets site tailoring information

Chapter 20

Troubleshooting your system

This chapter covers common problems that you may have while administering your system. The topics that are covered in this chapter include:

- Situations in which the system does not boot
- Reasons why the console and console keyboard do not work
- Common problems with filesystems, including repair and recovery
- Problems that occur during the installation procedure
- Situations where users cannot log into the system
- Solutions to common problems running **mail**
- Troubleshooting information for dial-out and dial-in modems
- Common problems with printers and the print system
- Information on stopping runaway and unkillable processes
- Starting stopped schedulers
- Recovering from other system failures, such as a power failure, system panic, or bad track on the hard disk
- Resolving security-related error messages
- Fixing common problems with tape drives
- Solving problems with terminals, such as fixing a scrambled display or unlocking a locked terminal
- Solving mouse problems.
- Troubleshooting the UUCP system

Solving startup (boot) problems

Situations in which the system does not boot can be divided into two categories:

- System does not boot during installation
- System does not boot even though it has successfully booted in the past

This section briefly describes common reasons why the system does not boot during installation. The discussion of what to do when your system does not boot after successful boots is more extensive. It contains instructions for recovering critical system files and solutions to specific problems, such as what to do when the system hangs at the login prompt and why the system cannot enter multiuser mode.

System does not boot during installation

If the system fails to boot from the floppy the first time during installation, one of several things may be wrong.

1. If the floppy in the drive is not the N1 (boot) floppy, the system does not display the boot prompt. Replace the floppy with the N1 floppy and reboot the machine.
2. If the floppy in the drive is the N1 floppy, make sure that the floppy is inserted correctly and that the floppy drive door is closed. Then, reboot the machine.
3. If you get a persistent read error on your N1 floppy, request a new floppy from your supplier.

If the system still does not boot, you most likely have a problem with your hardware.

1. Check your computer's documentation to see that your system has enough RAM (Random Access Memory). Your computer must have the minimum recognized memory listed in your *Release Notes* to install the operating system. If there is not enough RAM, the system cannot boot.

   At the boot prompt entering the command **mem=/p** displays all the RAM the system will use. On most systems you can override this default using the **mem=** command. See **boot**(HW).

   See the section "divvy: mount on /mnt Failed" under "Troubleshooting installation problems" later in this chapter for information on memory requirements.

2. If your system has enough RAM, check to see that the boards (bus cards) are seated properly in the motherboard.
3. If the system still does not boot, you may have a badly configured floppy drive.
4. If, after performing all the above tests, the system still does not boot, you may have a hardware failure. Check the documentation that came with your computer for hardware tests.

System does not boot after successful boots

If your system booted successfully in the past but does not boot now, the problem may be due to one of the following situations:

- The floppy drive contains a floppy that is not a boot floppy.
- Critical system files, such as */boot* or */unix*, are corrupted or missing. See the section "Restoring missing or corrupted system files" later in this chapter for information on booting from floppy disks and restoring these files.
- Your hard disk may have developed a bad track, corrupting system files that are required for booting the system. For information on how to recover from this situation, see the section "Mapping a bad track" under "Recovering from other system failures" later in this chapter.
- If your system does not have a 387 math coprocessor chip, and the */etc/emulator* file is missing, the system does boot, but a warning is displayed. If this is the case, see the section "Cannot load floating point emulator" later in this section.

Restoring missing or corrupted system files

On rare occasions, one or more of the critical system files is accidentally modified or removed, preventing the system from booting or operating correctly. In cases where your system does not boot, you must boot from floppy disks in order to access the system so that you can restore the critical files from backups.

In order to boot and access a system that does not boot from the hard disk, you must have made an Emergency Boot Floppy set as directed in the *Installation Guide.* This set consists of the boot floppy and the root filesystem floppy. The boot floppy contains three files necessary for booting and loading the UNIX system kernel: */boot, /etc/default/boot,* and */unix.* The root filesystem floppy contains a subset of the UNIX system utilities that you can use to restore your system.

NOTE You must have a separate Emergency Boot Floppy set for each system or further corruption can result.

If you have not made these floppies, you must reinstall the operating system. In some cases, if you do not also have a backup of the root filesystem, you must reinstall the operating system. To do this, follow the instructions in the *Installation Guide*.

/boot not found

If your system displays the following message when you turn on the power to your computer, the */boot* file is missing:

```
/boot not found
Stage 1 boot failure: error loading /boot
```

The */boot* file contains the **boot**(HW) program which loads and executes the kernel each time you turn on the computer.

If */boot* is missing, use the following procedure to boot the system from the Emergency Boot Floppy set so that you can then restore the file:

1. Insert the boot floppy in the drive and reboot the machine. This executes the initial boot from the boot floppy.
2. At the boot prompt, enter

   hd(40)unix

   After you boot the system from the floppy, this command loads the kernel from the hard disk.
3. Bring up the system in single-user mode by entering the root password at the prompt.
4. Mount the floppy filesystem from the command line using:

   mount /dev/fd0 /mnt
5. While the floppy is in the drive, restore */boot* by entering the following command at the system prompt:

   cp /mnt/boot /

   This allows you to access the */boot* file on the hard disk.
6. Before you remove the floppy disk from the drive, unmount the floppy filesystem (*/dev/fd0*) by entering:

   umount /mnt
7. Remove the floppy disk from the drive and bring down the system using **haltsys**(ADM).
8. Reboot the system from the hard disk by pressing ⟨Return⟩ at the boot prompt.

unix not found

If the system displays the following message when you turn on the power to your computer, the */unix* file is missing:

```
unix not found
```

The */unix* file contains the UNIX system kernel. If */unix* is missing, you can boot from another kernel file, such as */unix.old, /etc/conf/cf.d/unix,* or */etc/conf/cf.d/unix.old,* by specifying the complete pathname of the file at the boot prompt.

If there are no other kernel files on the system, use the following procedure to boot the system from the Emergency Boot Floppy set so that you can then restore */unix*:

1. Insert the boot floppy in the drive and reboot the machine.
2. At the boot prompt, enter

   fd(64)unix root=hd(40) swap=hd(41)

   This loads the kernel from the boot floppy and mounts the root filesystem on the hard disk.
3. Bring up the system in single-user mode.
4. Mount the floppy filesystem as */mnt*. (See step 4 of the previous section, "/boot not found.")
5. While the floppy is in the drive, restore */unix* with the following command:

   cp /mnt/unix /

   This copies the */unix* kernel file from the boot floppy to the hard disk.
6. Unmount the floppy filesystem. (See step 6 of the previous section, "/boot not found.")
7. Remove the floppy from the drive and bring down the system with the following command:

   init 6

   The root prompt (#) is returned, but the shutdown process begins soon afterward.
8. Reboot the system by pressing ⟨Return⟩ at the boot prompt.
9. You should relink the kernel at this point to ensure the correct devices are linked in.

Cannot load floating point emulator

If your computer does not have a 387 math coprocessor chip and the */etc/emulator* file is missing or corrupted, the boot fails with one of the following messages:

```
WARNING:  Cannot load floating point emulator
WARNING:  Zero length FP emulator file
WARNING:  Cannot read emulator file header
```

(If the 387 chip is present, the kernel recognizes it in the hardware recognition boot message.)

If the boot fails with this message, use the following procedure to boot the system and restore */etc/emulator*:

1. Insert the boot floppy in the drive and reboot the machine.
2. At the boot prompt, press ⟨Return⟩ and when instructed, insert the root floppy. This boots the system and mounts the root filesystem from the floppies.
3. Working from the floppy disk, use the following **mount**(ADM) command to mount the hard disk root filesystem to */mnt*:

   mount /dev/hd0root /mnt

   If **mount** fails, see the section "Filesystem mount failed" under "Fixing filesystem problems" later in this chapter for information on checking the hard disk with **fsck**(ADM).
4. Copy */etc/emulator* from the root filesystem on the floppy to the mounted hard disk:

   cp /etc/emulator /mnt/etc/emulator
5. Unmount the hard disk:

   umount /mnt
6. Make sure that the floppy is still in the drive and reboot the system with **haltsys**(ADM).
7. Remove the floppy from the drive and press ⟨Return⟩ at the boot prompt to boot from the hard disk.

System hangs at boot time

If the boot process hangs after the "Kernel: i/o bufs" message, the */etc/init* file is missing from the system. The */etc/init* file contains the **init**(M) program. Once started, the **init** process spawns all other processes on the system. Without the */etc/init* file, no new processes are started.

Use the following procedure to restore */etc/init*:

1. Insert the boot floppy in the floppy drive and reboot the machine.
2. Press ⟨Return⟩ at the boot prompt and when instructed, insert the root floppy.
3. Mount the hard disk root filesystem:

   mount /dev/hd0root /mnt

   If **mount** fails, see the section "Filesystem mount failed" under "Fixing filesystem problems" later in this chapter for information on checking the hard disk with **fsck**(ADM).
4. Copy the */etc/init* file from the root filesystem on the floppy to the mounted hard disk:

   cp /etc/init /mnt/etc/init
5. Unmount the hard disk by entering **umount /mnt**.
6. With the floppy in the drive, reboot the system with **haltsys**(ADM).
7. Remove the floppy from the drive and press ⟨Return⟩ at the boot prompt to boot from the hard disk.

System cannot enter multiuser mode

If the system fails to enter multiuser mode when you press ⟨Ctrl⟩**d**, or the following message is displayed at boot time, the */etc/inittab* file is missing:

```
INIT: Cannot open /etc/inittab errno: 2
INIT: SINGLE USER MODE
```

The */etc/inittab* file contains instructions for **init**. When *inittab* is missing, **init** cannot execute the system start-up instructions and the system cannot enter multiuser mode. When you press ⟨Ctrl⟩**d**, the system remains in single-user mode and displays the boot error message above.

As */etc/inittab* contains instructions that reference other parts of the system, a special */etc/inittab* is written to the root filesystem floppy when you create the Emergency Boot Floppy set. Because of this, the procedure for restoring your system is more complex than simply copying */etc/inittab* from the floppy filesystem to the hard disk.

To restore *inittab*, you must re-create the kernel environment.

To do this, use the following procedure:

1. Enter the following commands:

   cd /etc/conf/cf.d
   touch /etc/.new_unix
   ../bin/idmkenv

2. You see the following:

```
The kernel environment includes device node files and /etc/inittab.
The new kernel may require changes to /etc/inittab or device nodes.

Do you want the kernel environment rebuilt? (y/n)
```

   Enter y and press ⟨Return⟩.

3. The following is displayed:

```
The kernel has been successfully linked and installed.
        To activate it, reboot your system.

Setting up kernel environment
```

4. Enter the following command to reboot your system:

   init 6

   The root prompt (#) is returned, but the shutdown process begins soon afterward.

When the system reboots, the new */etc/inittab* file is in place.

/etc/bcheckrc not found

If the system displays the following message when you boot up, the */etc/bcheckrc* file is missing:

```
INITSH: /etc/bcheckrc: not found
```

The **init** utility executes **bcheckrc** according to instructions in */etc/inittab* whenever the system is booted. This utility checks the root filesystem and repairs it, if necessary. The */etc/bcheckrc* file should be on the hard disk when you boot the system. If */etc/bcheckrc* is missing, use the following procedure to recover it:

1. Bring up the system in single-user mode.
2. Clean the root filesystem manually with **fsck** before doing anything on the system:

   /etc/fsck -b /dev/root

3. Once the root filesystem is clean, make the following selections from **custom**:

   Install
   SCO UNIX System V Operating System
   Service Components
   SCO UNIX System V Runtime System
   Files
   UNIX Run Time System
   ⟨F5⟩
   bcheckrc

   Exit **custom** when the file is restored.

4. Enter **haltsys** at the prompt and reboot the system.

For more information on **bcheckrc**, see the manual page on **brc**(ADM).

execlp of /bin/sulogin failed

If the system displays the following message and goes directly into multiuser mode (run-level 2) at boot time, the */bin/sulogin* file is missing from the hard disk:

```
INIT: execlp of /bin/sulogin failed; errno = 2
```

The **sulogin**(ADM) utility must be present on the system to access single-user mode. If this file is missing, log in as *root* and make the following selections from **custom**:

Install
SCO UNIX System V Operating System
Service Components
SCO UNIX System V Runtime System
Files
UNIX Run Time System
⟨F5⟩
sulogin

Exit **custom** when the file is restored.

System hangs at login prompt

If the system boots correctly but hangs at the login prompt when you enter multiuser mode, the */bin/login* file is missing. The */bin/login* file contains the **login**(M) program. This command is run at the beginning of each terminal session to allow users access to the system. Follow these steps to restore */bin/login*:

1. Power-cycle the machine and press ⟨Return⟩ at the boot prompt.
2. At the prompt, enter the root password to go into single-user mode.
3. Make the following selections from **custom**:

   Install
   SCO UNIX System V Operating System
   Service Components
   SCO UNIX System V Runtime System
   Files
   UNIX Run Time System
   ⟨F5⟩
   login

   Exit **custom** when the file is restored.

Resolving console problems

This section covers some common problems that you may have with the console on your system:

- The keyboard on your console locks up
- How to prevent keyboard lockup by applying a special "patch"
- The console does not recognize keyboard input because you selected the wrong console keyboard type
- You cannot log into the console multiscreens when the system is in multiuser mode

Console keyboard locks up

When the system does not respond to input from the console keyboard, the situation is known as "keyboard lockup." Console keyboard lockup only affects keyboards that are attached to the video display adapter, not standard terminals that are attached to serial lines.

You may be experiencing keyboard lockup if the following statements are true:

- The system console keyboard cannot be used to enter data or perform any tasks.
- You cannot use the ⟨Alt⟩⟨F1⟩ through ⟨F12⟩ keys to switch multiscreens, and the ⟨CapsLock⟩ key does not turn the CapsLock light on or off.
- Other terminals on the system continue to work.
- Printers or other devices continue to work, and the system is still running.

Before trying to fix a locked keyboard, make sure that:

- You did not accidentally press ⟨Ctrl⟩s (which stops the screen from scrolling). To check this, press ⟨Ctrl⟩q and then see if you can enter characters from the keyboard.
- If your computer has a ⟨Keyboard Lock⟩ key, it is not in the locked position.
- The keyboard is plugged into the correct socket.
- The system itself is still running.

Check a terminal to see if it is still working and that you can perform system tasks, such as logging in and checking the date. If you do not have a terminal, watch the hard disk access light, if your computer has one. If it flashes periodically, at least once every 30 seconds, the system is still running and is using the hard disk.

NOTE You cannot use other terminals and the hard disk access light may not flash if you are in single-user mode.

If the console keyboard is still locked after checking these suggestions, unplug the console keyboard, then plug it in again. If this fixes the problem, your situation is definitely keyboard lockup. If this last step does not fix the problem, you may still have keyboard lockup.

Preventing console keyboard lockup

You can prevent keyboard lockup by applying a special "patch" that changes the operating system kernel. (The kernel is the main program of the operating system that is always running in memory.)

NOTE This patch disables the keyboard lights, so you should use it only if you have tried the other approaches.

The procedure for using the patch is as follows:

1. Get the system console working, if it is not. Reboot the system if you have to and bring it up in single-user mode.

   If you did not reboot, log in as *root* on the system console and shut the system down to single-user mode with the **shutdown** command:

   ```
   /etc/shutdown su
   ```

   (For more information, see the **shutdown**(ADM) manual page.)

2. Once the system is in single-user mode, back up the kernel with the following commands:

   ```
   cd /
   cp unix unix.00
   ```

3. Now, patch the kernel with the following commands:

   ```
   /etc/_fst -w /unix
   ledspresent/w 0
   $q
   ```

4. Shut down the system using **/etc/shutdown.**

5. When you see the "Normal System Shutdown" message, press any key to reboot the system. You have now fixed the keyboard lockup problem. You will need to reapply this patch each time the kernel is relinked.

Wrong console keyboard type

If your console keyboard is an XT or other non-AT keyboard and the operating system is configured for use with an AT keyboard, the system does not recognize input from the keyboard. For information on testing and switching keyboard modes, see the section on console keyboard type selection in the "Using the system console and color displays" appendix in this guide.

Cannot log into console

If you try to log into the console in multiuser mode, and the system displays the following error message:

```
Cannot obtain database information on this terminal
```

You should refer to the instructions for this error message found in the "Security-related error messages" section of the "Maintaining system security" chapter in this guide.

Fixing filesystem problems

This section describes the following:

- how to free up inodes
- how to create space when a filesystem runs out
- how to repair the filesystem when an abnormal shutdown corrupts the data
- how to check and repair a filesystem when **mount**(ADM) fails
- why a user is unable to remove files from a directory with the *sticky bit* set.

Out of inodes on filesystem

When a filesystem runs out of inodes, the system displays the following error message:

```
NOTICE:Out of inodes on dev nn/mm
```

where ***nn/mm*** is the filesystem that has run out of free inodes. To fix this problem:

1. Remove unnecessary (old, temporary, *core*, or log files) files from the filesystem.
2. Check to see if there is a large number of small files on the filesystem using the **find**(C) command with the **-size** parameter.

The number of inodes available on a filesystem is determined when the filesystem is created (using **mkfs**(ADM)). If the filesystem consistently runs out of free inodes, you can reconfigure the filesystem and increase the number of inodes. To do this:

1. Back up the filesystem using the **sysadmsh** selection:

   Backups ➪ Create ➪ Unscheduled

2. Verify the integrity of the backup using the **sysadmsh** selection:

   Backups ➪ Integrity

3. Unmount the filesystem using the **sysadmsh** selection:

   Filesystems ➪ Unmount

4. Run **mkfs** from the command line and specify more inodes for the filesystem.

   NOTE This procedure destroys the information on your hard disk. Do not use the **mkfs** command without a complete and verified backup.

For example, to reconfigure the number of inodes on the */dev/u* filesystem to 6400, use the following **mkfs** command:

```
mkfs /dev/u fssize:6400 9 400
```

where **9** is the gap, and **400** is the number of blocks/cylinder. ***fssize*** must be replaced with the size of the filesystem in question, in 1K blocks. This value can be obtained as described in "Repairing a filesystem when fsck stops at size check" later in this section. For more information on the **mkfs** command, see the **mkfs**(ADM) manual page.

5. Mount the filesystem using the **sysadmsh** selection:

   Filesystems ➪ Mount

6. Restore the filesystem from the backup with the following **sysadmsh** selection:

   Backups ➪ Restore ➪ Full

Out of space on filesystem

When a filesystem has little or no space left to work, the system displays the following message:

```
NOTICE:no space left on dev: nn/mm
```

When the filesystem runs out of space, the system stops any attempts to write to the filesystem. The only way to restore system operation is to delete or reduce files from the named filesystem.

Use the following suggestions to restore space to the filesystem:

1. Use the **wall**(ADM) command to send a system-wide message asking users to remove unnecessary files.
2. Check for the number of blocks used by each file and directory in the named filesystem with the **du**(C) command. For example, to display the number of blocks in the files in the */u* filesystem, use the following command from */u*:

   ```
   du | sort -nr
   ```

   This command sorts the files by size and displays them with the largest first. You can then send mail, asking the users who own the largest files to remove them.
3. Use the **find**(C) command to locate exceptionally large or old directories and files, and send mail to the owners asking them to remove unnecessary files.
4. Use **find** to locate and remove temporary files, *core* files and any unused *a.out* files. Remove files from the */usr/preserve* and *lost+found* directories. In addition, you can modify **cleantmp**(ADM) to define how often key directories (*/tmp* and */usr/tmp* by default) are cleared of files.

5. Clear the contents of system log files, such as */usr/spool/netmail/smtpc.log*, */usr/spool/lp/log*, */usr/adm/messages* and */usr/adm/sulog* (if enabled), and any log files for add-on programs.

   To do this, use the following construction:

   > ***filename***

6. If you have security auditing enabled, check the disk usage in the */tcb/audittmp/audit** directories. Back up and remove old audit files using the **sysadmsh** selections:

   System ➪ Audit ➪ Files ➪ Backup

   System ➪ Audit ➪ Files ➪ Delete

   You should check and remove these files regularly.

7. Reduce disk fragmentation by following the steps in the section "Reducing disk fragmentation" later in this chapter.
8. If the system is chronically short of free space, create and mount an additional filesystem.

Each of these steps is described in more detail in the section on maintaining free space in filesystems in the "Managing filesystems" chapter in this guide.

Checking free space on filesystems

You should periodically check the amount of free space in your filesystems with the **sysadmsh** selection:

System ➪ Report ➪ Disk

This command prints the amount of space left in the filesystem in *blocks* of 512 bytes. If the space is low, see the previous section for hints on what to do (before you run out of space).

Reducing disk fragmentation

If your system has been in use for some time, the constant creation and removal of files creates a situation called *disk fragmentation*. This means that the files in the filesystem are written in small pieces scattered widely across the hard disk. This results in increasingly poor disk I/O performance. Fragmentation becomes a problem when the disk is (approximately) more than 75% full.

NOTE The following procedure cannot be used on the root filesystem.

To reduce disk fragmentation, first make a complete backup of all the files in the filesystem:

1. Invoke **sysadmsh** and select:

   Backups ➪ Create ➪ Unscheduled

2. Select the filesystem and media device (block size is selected automatically) and format floppy disks, if necessary.
3. Load a volume, tape or disk, into the selected drive, and press ⟨Return⟩. The system copies files to the drive.
4. Next, verify the integrity of your backup. Invoke **sysadmsh** and select:

   Backups ➪ Integrity

   Enter the media type and insert each volume of the backup in turn.
5. Now, remove the files on the filesystem. For example, to remove the files from the */u* filesystem, enter the following commands at the prompt:

   ```
   cd /u
   rm -rf *
   ```

   NOTE This does not remove dot (.) files. To remove these files as well, change to the directory where the filesystem is mounted and enter the following command:

   ```
   find . -name ".*" -exec rm {} \;
   ```

6. Next, unmount the filesystem:

   ```
   umount /dev/u
   ```

7. Clean the filesystem with **fsck**(ADM):

   ```
   fsck -s /dev/u
   ```

8. When **fsck** exits, remount the filesystem:

   ```
   mount /dev/u /u
   ```

9. Finally, restore the files from the backup. Invoke **sysadmsh** and select:

   Backups ➪ Restore ➪ Full

   Insert the first volume and enter the name of the filesystem that you want to restore and the media device. The actual **cpio** command line is displayed.

Because the files are completely rewritten on the disk, each file is written in one piece and fragmentation is reduced. You recover a small amount of space (generally about 5 to 10 percent). It is a good idea to reduce disk fragmentation about once a year on a heavily-used system and less often on a lightly-used system. Verify that you have complete, accurate, and readable backups before you begin or you may lose files.

Restoring a corrupted root filesystem

If your root filesystem is so corrupted that, when you boot the system, **fsck**(ADM) cannot run, use the following procedure to restore your system:

1. Insert the boot floppy in the drive and reboot the machine.
2. At the boot prompt, press ⟨Return⟩. At the prompt, insert the root filesystem floppy. This boots the system and mounts the root filesystem from the floppies.
3. At the system prompt, enter:

   /etc/fsck -y /dev/hd0root

   You should see messages indicating that **fsck** is proceeding through five or six phases of system cleaning. If **fsck** exits within a few seconds or the system displays error messages that make no sense, you must restore the entire root filesystem from backups. Here are two examples of nonsense messages:

   ```
   UNKNOWN FILE SYSTEM VERSION 65535

   CLEANING NON SYSTEM 3 FILESYSTEM
   ```

   (Also see the following section, "Repairing a filesystem when fsck stops at size check.")

   If **fsck** appears to be successful, shut down the system with **haltsys**(ADM) and boot from the hard disk by pressing ⟨Return⟩ at the boot prompt. If **fsck** is not successful, repair the filesystem using **divvy** and restore from the backups. See "Repairing a filesystem when fsck stops at size check" for information on how to use **divvy**.

If you cannot boot from your hard disk at this point, you may have to reinstall the operating system from scratch. See the *Installation Guide* for more information.

Repairing a filesystem when fsck stops at size check

If the super block for a filesystem is so badly damaged after an abnormal shutdown or hardware failure that **fsck**(ADM) quits at the size check or reports an unusually large number of errors, you can use the **fsdb**(ADM) ("filesystem debugger") utility to patch the super block by hand.

The **fsck** utility reads the size of the filesystem from the super block. If the values currently stored in the super block for **FSIZE** (the total number of blocks in the filesystem) and **ISIZE** (the number of inodes allocated in the filesystem) are not the normal values for the filesystem, **fsck** displays the results of the size check and quits, as in the following example:

```
# fsck /dev/root
/dev/root
/dev/root File System: / Volume: root
Size check: FSIZE 0 ISIZE 0
#
```

When using **fsdb**, you have to change only one of the two values in the super block, **FSIZE** or **ISIZE**, back to their normal values for your filesystem.

NOTE You cannot run **fsdb** on the root filesystem if you are unable to boot up the system.

Because **fsdb** is a powerful tool that allows you to directly change the super block (which contains important information about your filesystem), be very careful when using this tool. If you enter improper values with **fsdb**, you can permanently damage your filesystem and lose all filesystem data. The **fsdb**(ADM) manual page describes a number of other ways in which you can use **fsdb** to look at and manipulate the super block. The use of such advanced features is recommended only for the experienced system administrator.

NOTE Make sure that the filesystem is unmounted before using **fsdb** to repair it.

To repair the damage described above, follow this procedure:

1. Enter the following command:

   divvy

   Because the **divvy** filesystem table is not located in the super block, the information in the table is probably correct. From the **divvy** table, obtain the first and last block number of the filesystem that you are attempting to restore.

2. To obtain the correct value for **FSIZE**, apply the following formula:

   FSIZE = last_block - first_block + 1

3. To obtain the correct value for **ISIZE**, run **bc**(C), and enter the following command:

   ((FSIZE * .25) - ((FSIZE * .25) % 16))

   where **FSIZE** is the value computed in step 2. Round down any fractions and record this calculation of **ISIZE** for later use. Exit **bc** by entering **quit**.

4. To convert the number of available inodes into the actual address of the first block following the blocks allocated for inodes, you must perform an additional calculation. Refer to this number as **isize** to distinguish it from the **ISIZE** value displayed when **fsdb** starts.

   To do this, perform the following calculation:

   isize=(ISIZE / 16) + 2

   where **ISIZE** is the number that you calculated in step 3.

5. With your **FSIZE**, **ISIZE**, and **isize** values, you can now correct the super block values with **fsdb**(ADM). Enter the following:

   fsdb /dev/*filesystem*

   where ***filesystem*** is the name of the filesystem that you want to repair.

6. If the values that you calculated are close approximations of the real numbers, **fsdb** displays output similar to the following:

   ```
   FSIZE = 52985, ISIZE = 13232
   ```

   If the match is close, you do not need to modify that parameter. If both values seem correct, something else is wrong and you must restore your data from backups.

   NOTE You can end the display of addresses at any time by pressing INTERRUPT or ⟨Del⟩.

7. Decide which parameter (or both), **ISIZE** or **FSIZE**, must be corrected.

8. Enter the following commands (for XENIX, or UNIX system and AFS filesystems), substituting the values that you calculated earlier for **FSIZE** and **isize**. In the instructions that follow, your input is in bold and **fsdb** responses are in normal font, with real values in place of xxx and yyy. Remember to use the actual values for **FSIZE** and **isize**; do not enter the words FSIZE and isize.

   If **isize** is correct, simply skip it by pressing ⟨Return⟩.

   NOTE Each time you press ⟨Return⟩, **fsdb** displays what the filesystem currently thinks the values of **FSIZE** and **isize** are.

To repair a XENIX filesystem, enter these commands:

```
1024<Return>
002000:  000000    (0)
=isize<Return>
002000:  000xxx
<Return>
002002:  000000    (0)
=FSIZE<Return>
002002:  000yyy
q
```

To repair an S51K, EAFS or AFS filesystem:

```
512<Return>
001000:  000000    (0)
=isize<Return>
001000:  000xxx
<Return>
001002:  000000    (0)
<Return>
001004:  000000    (0)
=FSIZE<Return>
001004:  000yyy
q
```

9. If the corrupted **FSIZE** value is so small that **fsdb** thinks that you cannot move that far into the block, the following error message is displayed when you input the beginning address:

   ```
   block out of range
   ```

   To disable error checking, enter a capital **O**. You can then input the address without complaint.

At this point, you can run **fsck** on the filesystem and properly restore the system.

NOTE Currently, you cannot use **fsdb** to repair a DOS filesystem.

Filesystem mount failed

If the **mount**(ADM) command fails, check the filesystem with the **fsck**(ADM) command before running **mount** again:

/etc/fsck -y /dev/*filesystem*

where *filesystem* is the name of the filesystem that you want to check and repair.

Unable to remove files

If a user has write permission on a directory but is unable to remove files from that directory, the *sticky bit* has been set for that directory. The sticky bit is a directory protection setting that allows only the owner of the file (or *root*) to remove files from that directory.

NOTE The contents of the files in the directory can be modified or removed, if the permissions allow write access.

Only *root* or the owner can set the sticky bit; only *root* or the owner of the directory can remove it.

To determine whether a directory has the sticky bit set, do a long listing of its parent directory (the sticky bit shows up as a "t" in the last field of the permissions listing). The following example shows a directory on which the sticky bit has been set:

```
drwxrwxrwt  5 sys     sys         2432 Jan 17 16:58 tmp
```

To remove the sticky bit setting from a directory, enter the following command:

chmod -t *directory_name*

Users should now be able to remove files from the directory.

Troubleshooting installation problems

This section discusses some problems you may have during initial installation or subsequent reinstallations of the operating system, minimum memory requirements, and how to restart a reinstallation.

divvy: mount on /mnt failed

If installation fails and the system displays the following messages, your system does not have enough memory:

```
divvy:mount on /mnt failed
invalid argument
cannot set up /dev directory on new device
```

UNIX System V/386 requires the minimum memory listed in the *Release Notes* to install. In addition, when you are using a minimal memory configuration, the memory must be fully-recognized. During installation, much of this memory is used as a RAM swap device. If you do not have enough memory, the system runs out of swap space during the filesystem creation procedure and installation fails.

If your machine has a minimal memory configuration, verify that the operating system recognizes all the memory you have installed. There are two ways of checking this. First, use the **mem=/p** boot command to show the actual RAM layout, and second, check the boot screen message. The system displays a message, like the following, after the hardware recognition information:

```
    mem:  total = 7808k, kernel = 2468k, user = 5340k
```

On some 386 machines, the UNIX system kernel cannot recognize the memory between 640 and 1024K because the hardware manufacturer has mapped this 384K of memory to another location for their firmware to use. Because this location varies, the UNIX system kernel does not know where to find it. The **mem=** boot command can be used to tell the system where the memory is.

To install the system and avoid the **divvy** error, add enough memory so that the UNIX system kernel recognizes the minimum required memory. We strongly recommend that you use 32-bit memory made by the manufacturer.

Use one of the following two methods to add memory:

- If the motherboard has only 640K, buy an expansion board and start it at 1024K. This is the best solution because no memory is "wasted."
- If the motherboard already has two megabytes and has additional empty sockets, add at least another 384K to the motherboard. If the motherboard does not have any empty sockets, you must to buy and install an expansion board.

See the "Adding multiport cards, memory, and other bus cards" chapter in this guide for more information on adding additional memory.

N1 disk boots from the hard disk

If you try to reinstall and the system boots from the N1 floppy but roots from the hard disk, you can restart the installation process by entering **restart** at the boot prompt.

Fixing login problems

This section covers what to do in situations in which:

- you cannot log in to multiscreens on the console after the system enters multiuser mode
- users cannot log into the system
- the system displays "Login Incorrect" when a user tries to log in
- the user's account is locked
- a user forgot their password

Cannot log in after entering multiuser mode

After entering multiuser mode, you cannot log into the console, even as *root*, and the system displays the following error message:

```
Cannot obtain database information on this terminal
```

Refer to the instructions for this error message found in the "Resolving security-related error messages" section of this chapter.

Users cannot log into the system

When the */etc/group* file is missing, users cannot log in and the system displays the following message on the user's terminal:

```
Can't rewrite terminal control entry for tty01.
Authentication error; See Account Administrator
```

The following error message is also displayed at boot time:

```
/bin/su:cannot setgid to auth, no auth entry
```

Refer to the instructions for this error message found in the "Resolving security-related error messages" section of this chapter.

Login incorrect

If the system displays the "Login Incorrect" error message when a user tries to log in, one of several problems may exist. Refer to the instructions for this error message found in the "Resolving security-related error messages" section of this chapter.

useshell: file access control database inconsistency

If the system displays the messages regarding "useshell" when creating or modifying a user, permissions on certain system files are incorrect. Refer to the instructions for this error message found in the "Resolving security-related error messages" section of this chapter.

Unlocking a locked user account

If a user account has been locked deliberately by the system administrator, the system locked the account because a user exceeded the number of unsuccessful logins attempts, or the user's password is dead, the system displays the following message when a user tries to log in to that account:

```
Account is disabled -- see Account Administrator
```

To unlock the account, invoke **sysadmsh** and select:

Accounts ➪ User ➪ Examine:Logins

Change the "Lock status" field to "Clear all locks". For more information, see "Locking or unlocking a user account" in the "Administering user accounts" chapter of this guide.

Replacing a forgotten user password

The system does not provide a way to decipher an existing password. If a user forgets their password, the system administrator must change the password to a new one.

To do this, follow these steps:

1. Invoke **sysadmsh** and make the following selection:

   Accounts ➪ User ➪ Examine:Password

2. Select "Change" from the Current password status selections.
3. Choose "Yes" from the Confirm Change form to select a new password. This invokes the password change procedure.
4. Select "1" (to select a password) or "2" (to have the system generate a password) and enter or choose a new password.
5. Press ⟨Return⟩ to return to the modify user account form.

For more information, see the "Administering user accounts" chapter of this guide.

Fixing mail problems

The MMDF system is configured for the local system by default. Therefore users should not experience problems with local mail. Generally, when **mail** does not work, there is either a problem with the network, or with MMDF configuration.

When mail problems occur, first verify that the network is working. To do this, test the appropriate network (for example, UUCP or TCP/IP). See the "Building a remote network with UUCP" chapter in this guide and the "Troubleshooting network connections" section in this chapter for information on testing the UUCP connection. See your networking documentation for other testing information.

If UUCP is working and you still experience problems with **mail**, the problem is most likely with the MMDF configuration or the **deliver** daemon. The following sections discuss some common problems and solutions.

Failed mail error

If the system returns your mail immediately in a mail message with the words "Failed Mail" in the Subject line, there is a problem with the channel (**.chn*) or domain (**.dom*) files in the */usr/mmdf/table* directory. Use the following procedure to check this:

1. Log in as the **mmdf** user.
2. Change directories to */usr/mmdf/bin*.
3. Invoke **checkaddr** with the mail address that you used. For example, if **mail sylvia@seattle** failed, enter:

   ```
   ./checkaddr sylvia@seattle
   ```

4. If **checkaddr** displays `Unknown Host Domain`, either the host or domain is not in the channel or domain files, or the */usr/mmdf/mmdftailor* file contains an error.

   If **checkaddr** displays "OK," then the host and domain are in the channel and domain files, but are incorrectly configured. Use **checkaddr -w** to diagnose this problem.
5. Using the information in the "Setting up electronic mail" chapter in this guide, verify that your domain, channel, and */usr/mmdf/mmdftailor* files are set up correctly.

Mail does not work, no returned mail

If mail does not reach its destination but MMDF does not return unsent mail, either **deliver** is not running properly, or the */usr/mmdf/mmdftailor* file contains an error. The **deliver** daemon is started automatically by a script in the */etc/rc2.d* directory at system startup; the process is owned by **mmdf**. Check to see that **deliver** is running using **ps -ef**. If **deliver** is not running, use the following procedure:

1. Log in as the **mmdf** user.
2. Start the **deliver** daemon manually with the following commands:

   ```
   cd /usr/mmdf/bin
   ./deliver -cchannelname -b -T60
   ```

   where *channelname* is the channel that you want to check (for example, UUCP or SMTP). This starts **deliver** for the specified channel sweeping the mail queue in the background every 60 seconds. For more information, see the **deliver**(ADM) manual page.

3. If running **deliver** manually does not work, check the mail queue with the following command:

   ./checkqueue -c channelname

   This command displays the messages in the mail queue for the local channel that have yet to be delivered.
4. If there are messages in the queue, run **deliver** again.
5. If running **deliver** manually again does not work, check the configuration of the domain and channel files in */usr/mmdf/table* and the */usr/mmdf/mmdftailor* file.

Mail command hangs

If the **mail** command hangs when a user tries to read mail, check the following:

1. Verify that the user's mailbox exists in the */usr/spool/mail* directory. Unless your postmaster has configured the system otherwise, the */usr/spool/mail* directory should contain a file for each person using the **mail** command.
2. Make sure that the owner and group ID of the */usr/spool/mail* directory is **mmdf.**
3. Make sure that the mailbox is not too large (1000 messages is regarded as too large).

Inconsistencies in MMDF system name

If you did not change the system name during installation, the system name that the MMDF mail system recognizes is different than the name that the **uname**(C) command reports and that the */etc/systemid* file contains. The MMDF system name causes mail to be addressed from **user@unix.UUCP** while **uname**(C), */etc/systemid,* and the login prompt all report the system name as *scosysv.* This causes inconsistencies when setting up networks such as UUCP, MicNet, and TCP/IP.

To solve these problems, change the system name:

1. Choose a new system name.
2. Log in as *root.*
3. Enter the following command at the prompt:

   uname-*S *newname
4. Log in as the **mmdf** user.
5. Edit the */usr/mmdf/mmdftailor* file, changing all occurrences of the word "unix" to the new name.

6. Edit the */usr/mmdf/table/*.dom* and */usr/mmdf/table/*.chn* files, changing all occurrences of the word "unix" to the new name.
7. Change to the */usr/mmdf/table* directory and enter:

   ```
   ./dbmbuild
   ```
8. Reboot the system.

The system name is now set correctly.

Troubleshooting your modem

This section discusses common problems that you may have with your modem. Note that, while other serial ports are often used, the examples in the modem troubleshooting sections assume that the modem is attached directly to COM1. If you have problems, first verify that the phone jack is plugged in and that you have a dial tone on the phone line.

Errors when dialing out

This section describes some situations and solutions to problems that may occur when dialing out on your modem.

The most useful tool for diagnosing dial-out problems is the **-x9** option to **cu**(C). This option causes **cu** to display diagnostic output when attempting to dial out. To get a debugging output, enter the command:

cu -x9 ***phone_number***

where ***phone_number*** is the phone number of the system you wish to dial.

No OK message

You get a connected message when you test the modem connection with the command:

```
cu -s1200 -l tty1a dir
```

but, when you enter **AT**, the system does not display the "OK" message. Use the following steps to solve the problem:

1. Verify that the modem switch and software settings are correct.
2. Check the modem cable:
   - If you are using a straight-through cable, try a null modem cable using at least pins 2, 3, 7, 8, and 20.
   - After issuing the **cu** command, watch the lights on the modem and press ⟨Return⟩ several times. The "receive" light should flash as you hit ⟨Return⟩. If it does not flash, check your cable to make sure that pin 2 is connected correctly (pin 2 is the data transmission line from the serial port to the modem).

3. If the "send" light flashes on the modem, the local echo may be turned off. Use the **ATE1** command to turn on the modem's echo capability.
4. Verify that the serial port on the computer is not defective:
   - Attach the modem to a different serial port, or attach a terminal or serial printer to the port to confirm that it is functioning.
   - If the port is not functioning, check your hardware documentation for an appropriate repair facility.
5. If the previous steps do not fix the problem, your modem may be defective. If this is the case, check your hardware documentation for an appropriate repair facility.

Modem dials, but does not connect

If your modem dials okay, but the call never connects, check the following:

1. Verify that the phone number is correct and operational and that the phone line to which the modem is attached is not faulty. To do this, unplug the modem from the telephone line and plug in a regular telephone. Manually dial the number to make sure that the modem on the other end of the line is answering the call.
2. Listen carefully to your modem while it dials the call. Some business phone systems require a pause between certain numbers. Use a hyphen in the **cu** command to indicate a pause of two seconds. In this example, the modem pauses 8 seconds after dialing the first number: **9----458--1234.**

   The dialer translates the hyphen passed to the **cu** command into the appropriate code for your modem. For example, the dialer translates the hyphen into a comma before sending to a Hayes-compatible modem.

Connect failed: NO DEVICES AVAILABLE

When you try to dial out on the modem, the following message is displayed:

```
Connect failed: NO DEVICES AVAILABLE
```

Follow these steps to solve the problem:

1. Verify that the modem port has an entry in the */usr/lib/uucp/Devices* file. Here are example entries for a Hayes-compatible modem running at 2400 baud on */dev/tty1A*:

   ```
   ACU  tty1A - 300-2400 /usr/lib/uucp/dialHA24
   Direct  tty1a - 2400 direct
   ```

   Make sure that these lines in *Devices* do not begin with a number sign (#). There should be no spaces in front of the entries.
2. Verify that the modem port in *Devices* has the correct baud rate associated with it. If you specify the baud rate with the **-s** option to **cu**, verify that there is an entry in *Devices* that corresponds to that baud rate.

Modem answers, but terminal displays garbage

If the modem answers, but the terminal displays garbage characters, use the following steps:

1. Verify that the site that you are calling is set to the same data bit and parity values that you are using. By default, **cu** uses 8 data bits, and no parity. To change the values to 7 data bits and even parity, enter **cu -e**. For 7 data bits and odd parity, use **cu -o**, and use **cu -oe** for 7 data bits and no parity.
2. Verify that the remote computer is set to the same baud rate that you are using.

   If you are dialing into another UNIX system, you can force the remote site to switch to the next lower baud rate by sending a break signal. Always start at the highest baud rate and move down as necessary. To send the break signal during the login sequence, enter:

   ```
   ~%b
   ```

3. Check for noise on your phone line. Noise problems become more acute when operating at 2400 baud or higher. Normally, when there is a problem with line noise, garbage characters appear on the screen continually, as if a system on the other end of the line is trying to send valid data.

DEVICE LOCKED

This message is described under "UUCP log and status file messages" in the "Troubleshooting network connections" section of this chapter.

Modem does not hang up

If your modem does not hang up at the end of a call, check the following:

1. Verify that you are using a modem control port that is configured in the */usr/lib/uucp/Devices* file. If you are using a non-modem control port, change the port to the corresponding modem control port. For example, the modem control port associated with tty1a is tty1A.

   NOTE Non-modem control ports should only be used with terminals, and when configuring the modem.

2. If the CD (Carrier Detect) light on the modem does not go off when the call is disconnected, check the modem switches to verify that the modem is set to detect the incoming carrier. If your modem is a Hayes 2400 or compatible, use the **AT&C1** command. This forces the carrier detect line to follow the presence of a carrier on the phone line.

3. Check the modem switches to verify that the modem is set to detect DTR (Data Terminal Ready). The modem should hang up when DTR goes from high to low. If the modem is a Hayes 2400 or compatible, use the **AT&D2** command.
4. Some modems have a switch that can be set to ignore DTR; make sure that this switch is off.

Double echo

If you get a double echo when you dial out on your modem, check the setting for local echo. If local echo is enabled, disable it.

Problems dialing in

This section provides solutions for common problems that may occur when dialing in on your modem.

Modem does not answer the phone

If the modem does not answer the phone, check the following:

1. Verify that the modem control port is enabled. To enable the modem port, enter the following commands:

   ```
   disable /dev/tty1a
   enable /dev/tty1A
   ```

2. Verify that the modem is configured to auto-answer. Check your modem switches. If the modem is a Hayes 2400 modem, enter:

   ```
   cu -l tty1a dir
   ```

   Then, use the **ATS0=1** command to tell the modem to answer the phone on the first ring. (Remember to enter **AT&W** to save modem settings.)
3. Verify that the DTR (Data Terminal Ready) line is connected from the computer to the modem. Make sure that Pin 20 is connected. Pins 2, 3, 7, 8, and 20 are required for modem communication.
4. Make certain the ACU entry for this modem in the **Devices** file precedes any Direct entries for the port or the non-modem control counterpart.

Modem answers, but hangs up

If the modem answers, but hangs up immediately upon connection, check the following:

1. If the modem is set to auto-answer and to detect DTR, check to see that the DTR line is asserted.

2. Verify that the modem control port is enabled:

   disable /dev/tty1a
   enable /dev/tty1A

3. Verify that the cable is correct. If you are using a straight-through cable with at least pins 2, 3, 7, 8 and 20 connected, verify that pin 20 (DTR) is properly connected.

Garbage or loose cable

The console displays a message like the following when a call comes into the modem:

```
Garbage or loose cable on /dev/tty1A, port shut down
```

Check the following:

1. Verify that your modem is not set to echo back data or send command responses. If the modem is not set up this way, it may be sending a "RING" signal to indicate that the phone you are calling is ringing. Because the CD signal is not active, getty interprets this as random data on the serial line. The proper Hayes 2400 modem command is **ATE0Q1**.
2. If you have an internal modem and the above options do not eliminate the error message, your modem may be incompatible. Replace your modem with a standard Hayes-compatible external modem.

Modem answers, but no login prompt

If the modem answers, but does not display a login prompt, check the following:

1. Verify that the CD line is being asserted by the modem after the modem has answered the phone. Check the switches on your modem or, if your modem is a Hayes 2400 or compatible, use the **AT&C1** command. (Remember to enter **AT&W** to save modem settings.)
2. Make sure that the port is enabled. Enable the port by entering the following command:

   enable /dev/tty1A

3. Verify that the modem is using the correct */etc/gettydefs* entry and is selecting the proper baud rate. The modem port device line in the */etc/inittab* file should look like the following:

   ```
   t1A:2:off:/etc/getty -t60 tty1A 3
   ```

   The last character on the line is the pointer to the entry in the */etc/gettydefs* file. In this case the */etc/gettydefs* entry is **3**, which is the 2400 baud entry. Verify that this entry in */etc/gettydefs* is correct.

Screen displays a series of login prompts

If the screen scrolls uncontrollably when you log in, usually displaying a series of login prompts, verify that only the modem device is enabled. If the non-modem device is enabled, disable it:

disable /dev/tty1a

System displays meaningless characters

If the system displays the login prompt, but no password prompt, or meaningless characters are displayed after the login prompt, verify that the line settings are correct:

1. Determine the serial line settings on the system that you are calling. The standard settings that **cu** uses are 8 data bits, one stop bit, and no parity.
   - If the remote system uses even parity, use **cu -e**.
   - If the remote system uses odd parity, use **cu -o**.
2. If you are dialing into a UNIX system, check the */etc/inittab* file on the remote system to verify that the "pointer" into the */etc/gettydefs* file is correct. The serial line characteristics may not match between the stty settings defined in the third field of the selected *gettydefs* entry. Change the setup for the port to 8 data bits, one stop bit, and no parity.

   The entry should look like this:

```
3 # B2400  HUPCL OPOST CR1 ECHOE NL1 #
        B2400 CS8 SANE HUPCL TAB3 ECHOE IXANY #\r\n@!login: # 1
```

Troubleshooting the print system

This section covers some common problems that you may have with your printer system:

- The printer does not print
- The system does not recognize the printer port at bootup
- Redirecting output to the printer port does not work
- The output from the printer is illegible
- The spacing on the output from the printer is wrong
- A parallel printer is abnormally slow
- A dialout printer reports UUCP errors

Printer does not print

If the printer is sitting idle and there is no output, check the following:

1. Make sure that the printer has power.
2. Verify that the printer hardware is working before continuing. Check the printer documentation for a self-test.
3. Check the printer cable and make sure that it is attached properly to the port and the printer. Refer to the owner's manual for your printer for installation instructions.
4. Make sure that the printer is configured properly. To set up your parallel or serial printer to receive data properly, follow the instructions in the section, "Installing a printer" in the "Using printers" chapter of this guide.

   If the printer is a serial printer, make sure that the baud rate at which the computer sends data to the printer matches the printer's baud rate. For instructions on how to reset the baud rate, see the section "Printer output is illegible" later in this section.
5. Make sure that the printer is enabled. To do this, invoke **sysadmsh** and select:

   Printers ⇨ Schedule ⇨ Accept

   Printers ⇨ Schedule ⇨ Enable
6. Verify that the system recognizes the port at boot time. After the copyright information, the system should display information like the following for each port (examples are for parallel and serial, respectively):

   ```
   parallel 0x378-0x37A   07    -    unit=0
   %serial 0x03F8-0x03FF 04 - unit=0 type=Standard nports=1
   ```

   If the system does not display a similar message for the printer port, follow the instructions in the "Port not recognized at bootup" section later in this chapter.
7. Make sure that the port is configured for the proper interrupt vector and that no other hardware is using that interrupt vector. For information on the available interrupt vectors, see the section "Tape not recognized at bootup" under "Troubleshooting your tape drive" in this chapter. See your hardware documentation for information on configuring your ports.
8. Test the printer port connection by redirecting the output of a command directly to the device.
   - For parallel printer **lp0**, enter:

     date > /dev /lp0

- For serial printer **tty1a**, enter:

  (stty *options*;date) > /dev/tty1a < /dev/tty1a

  where ***options*** are baud rate, parity, or other settings that you want to pass to the serial printer.

If the output from the redirected command does not print, follow the instructions in "Cannot redirect output to printer" in this section.

If the output prints, try submitting a sample file (like */etc/motd*) for printing by invoking **sysadmsh**, and selecting:

Dirs/Files ⇨ Print

If the hardware connections are good and the printer is properly configured and enabled, but is still idle and print requests are queued, check the following:

1. Verify that the **lpsched** process is running:

   ps -u lp

2. Restart the **lpsched** daemon if it is not present:

   /usr/lib/lpshut
   /usr/lib/lpsched

3. Check to see that print requests are being filtered:

   lpstat -o -l

   This command displays a detailed description of the status of output requests, printer names, and devices.

4. If the printer detects a fault, it does not immediately continue automatic printing. Force a retry by enabling the printer.
5. Check to see if a dialout printer was busy or did not answer, or all dialout ports are busy. The print service waits five minutes before trying to reach a dialout printer again. Force a retry by enabling the printer.

Cannot redirect output to printer

If you redirect output directly to the parallel or serial port and nothing happens or the system displays the "cannot create" message, check the following:

1. Verify that the device file for the port exists in */dev*. Make sure that this file is a device file and not a text file. For example, use the following command to check **lp0**:

   ls -l /dev/lp0

2. Test the cable connection using a cable from a working system.

3. Print a file from DOS. If you can print a file under DOS but not under the UNIX system, check the following:
   - Verify that the port is recognized at boot time and that it is configured correctly.
   - If the port configuration is correct, and you still cannot redirect output to the port, try using a different device name. For example, for a parallel port, use **lp1** instead of **lp0**; for a serial port, use **tty2a** instead of **tty1a**.
   - If you still cannot print using a different device name, your printer may be defective; check the hardware documentation that came with your printer.
4. If you cannot print from DOS, check the printer hardware configuration. See the documentation that came with your printer.

If you configured your printer correctly and you still cannot redirect output to it, the problem is most likely a hardware malfunction. Recheck the cables and port configuration and consult your hardware documentation.

Port not recognized at bootup

If your system does not recognize the port at boot time (the "parallel" or "serial" line for your port is not displayed after the copyright information), check the following:

1. Verify that the parallel or serial card is properly seated:
   - Turn the power off and open the machine.
   - Remove and reseat the card in the bus.
2. If you have more than one parallel card, one may be conflicting with the other; remove the second card.
3. Verify that the card is correctly configured; check the documentation that came with the card. If possible, try setting the card for a different configuration.
4. The card may be defective; replace it.

Printer output is illegible

If the printer prints illegible output, check the following:

1. Determine the baud rate for the serial printer and check to see that it matches the baud rate for the computer. (If the printer is connected with a parallel port, the baud rate does not matter.) Set the baud rate to 9600 baud for optimum performance.

To set a different baud rate for the print service to use, use the following command:

Printers ➪ Configure ➪ Parameters

In the "Default initial settings" section, enter the baud rate number in the "stty" field. Then submit a sample file for printing.

2. Determine the parity setting for the printer and check to see that it matches the computer's parity setting. (If the printer is directly connected to the computer with a wire that is less than 50 feet long, it does not have to use the parity bit.)

   To set the parity bit, use the following **sysadmsh** selection:

   Printers ➪ Configure ➪ Parameters

   In the "Default initial settings" section of the form, add one of the following to the "stty" field:

oddp	Sets odd parity generation.
evenp	Sets even parity generation.
-parity	Sets no parity (default).

   Select the option that matches what your printer requires.

3. Check to see that the tabs are set correctly. See the next section "Printer output spacing is wrong."

If the settings and baud rate are correct and the output is still illegible, check to see that the printer type is correct. If the wrong printer type was selected when you set up the printer with the print service, the wrong control characters can be sent to the printer. This situation can cause output to disappear or be illegible. A simpler problem to solve is when it sets the wrong character set or font.

If you do not know the printer type, use the following steps to examine the available printer types:

1. If you think the printer type has a certain name, enter the following command at your system prompt:

   TERM=*printer-type* tput longname

   A short description of the printer identified by ***printer-type*** appears on your terminal. Try names in place of ***printer-type*** until you find one that identifies your printer.

2. If you do not know what names to try, use the following command to examine the */usr/lib/terminfo* directory for a list of names that are available:

   ls -R /usr/lib/terminfo | more

   Choose names from the list that match one word or number identifying your printer. For example, the name 495 identifies the AT&T 495 Printer. Try each of the names in place of ***printer-type*** in the command in step 1.

When you have the name of a printer type that you think is correct, change the printer type setting. To do this:

1. Invoke **sysadmsh** and select:

   Printers ⇨ Configure ⇨ Parameters

2. In the printer name (terminfo database) field, enter the printer type name.

Printer output spacing is wrong

If the printer output is legible, but the spacing is wrong, invoke **sysadmsh** and select

Printers ⇨ Configure ⇨ Parameters

For each of the situations, adjust the following settings in the "stty" field of the "Default initial settings" section:

- If the printer output is double-spaced, enter either the **-onlcr** or **-tabs** option.
- If there is no left margin and the text runs together, enter the **-tabs** option.
- If the printer output zig-zags down the page, enter the **onlcr** option. (This is set by default, but you may have cleared it accidentally.)

Parallel printer is slow

If your parallel printer prints abnormally slowly, verify that the configuration settings are correct using the section "Installing printers" in the "Using printers" chapter of this guide.

If printing is still slow after verifying that the parallel ports are configured correctly, the problem may be that your parallel port is not capable of generating interrupts.

To speed up printing on your parallel printer, you can alter the way that the hardware and the printer driver communicate. The parallel printer driver can be made to "poll" a parallel port so that the driver does not rely on interrupts from the parallel port.

NOTE When the printer driver polls a parallel port, you may experience a drain on system resources.

To set up polling for a parallel port or parallel printer, create what is known as a **special device node**.

Use the following procedure:

1. Note which parallel printer ports are recognized during the bootup message. (You can also look at the last boot message in */usr/adm/messages*). For example:

   ```
   parallel 0x378-0x37A   07    -    unit=0
   ```

   where "unit=0" refers to *lp0*.
2. Log in as *root*.
3. Create a special device file for the printer by entering one of the following commands:
   - For **lp0**, enter: **mknod /dev/lp0p c 6 64**
   - For **lp1**, enter: **mknod /dev/lp1p c 6 65**
   - For **lp2**, enter: **mknod /dev/lp2p c 6 66**
4. Enter the following commands for each printer that you want to configure. For example, enter the following for */dev/lp0*:

   chown bin /dev/lp0p
   chgrp bin /dev/lp0p
   chmod 222 /dev/lp0p
5. If you are using the print spooler, you must now inform the spooler of the new parallel poll device. Invoke **sysadmsh** and select:

   Printers ⇨ Configure ⇨ Modify
6. In the "Device name" field, do not select a standard parallel device name. Instead, use either */dev/lp0p*, */dev/lp1p*, or */dev/lp2p*.

Printer reports UUCP errors

If UUCP is configured, the print service uses the UUCP software to handle dialout printers. If a dialing failure occurs and you receive printer fault alerts, the print service reports the same error reported by the UUCP software for similar problems. (If you have not arranged to receive fault alerts, this information is mailed to the user **lp**, by default.) For more information, see the UUCP error messages in the "Building a remote network with UUCP" chapter of this guide and the section "Troubleshooting network connections" later in this chapter.

Resolving process errors

This section discusses how to recover from situations in which the system runs out of processes (the process table is full), a runaway process locks up a keyboard, or a process is unkillable. This section also covers how to reconfigure the maximum number of processes systemwide and per user.

No more processes

The number of simultaneous processes that each (non-super) user can have is limited. Each process running on the system uses one entry in the kernel process table. If an attempt is made to create a new process when there are none available (when the process table is full), or when the process limit for a UID is exceeded, the **fork** fails and the system displays the following error message:

```
No more processes:
```

This error produces no console messages.

When the system displays this error message, use the **ps**(C) command to check for a *runaway process.* To display all the processes on the system, enter:

ps -ef | more

For more information on this command, see the **ps**(C) manual page. The system can run out of available processes when a program enters an infinite loop and spawn new processes. If this is the problem, refer to the section "Stopping a runaway process" later in this chapter.

If the system consistently displays the "No more processes" error message, use the following procedure to evaluate whether your system needs tuning:

1. Use the **crash** command to determine the current size of the process table:

   You see the following display:

   ```
   dumpfile= /dev/mem, namelist= /unix, outfile= stdout
   >
   ```

   At the " > " prompt, enter **var** and press ⟨Return⟩.

   A list of system variables is displayed; "v_proc" is the maximum number of processes that can exist on the system at one time.

2. Determine the number of processes currently running when the system is under heavy load with the following command:

   pstat -p | head -1

3. The value for **NPROC** (which is the same as v_proc) should be at least 10% greater than the number of processes running at peak use time.

If the **NPROC** value is too small, you should increase the value and thus reconfigure the kernel to allow for more processes. To increase the maximum number of processes per user, you should modify the **MAXUP** parameter.

For information on tuning system parameters, refer to "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide.

Stopping a runaway process

A *runaway process* can cause an error that locks up the keyboard, preventing anything that the user types from reaching the system. Because of this, a runaway process cannot be stopped from the terminal at which it was invoked.

To continue, you must identify and stop the runaway process:

1. Log in as *root* on a terminal (or console screen) that is not locked.
2. Enter the following command and press ⟨Return⟩:

   ps -ef

   The system displays full information on all the current processes and their process identification numbers (PIDs). Find the PID of the runaway program.
3. Enter

   kill *PID*

   The program should stop in a few seconds. If the process does not stop, enter

   kill -9 *PID*

The last command may leave temporary files that are usually removed when a program terminates normally, or a non-echoing terminal. To restore the terminal to normal operation, follow the instructions in the section "Restoring a non-echoing terminal" under "Solving terminal problems" later in this chapter.

Removing an unkillable process

If the runaway process does not stop when you enter **kill** with the **-9** option, the process is considered *unkillable.* If the unkillable process is a user's shell, you must stop the process before that user can continue working. To stop an unkillable process, you must reboot the system. Use the following procedure:

1. Log in as *root* and send a message using **wall**(ADM) to the other system users notifying them of the impending shutdown.

2. When all the users have logged out, shut down the system by entering **init 6**. The root prompt (#) is returned, but the shutdown process begins soon afterward.
3. Reboot the system by pressing ⟨Return⟩ at the boot prompt.

The unkillable process no longer appear when you run **ps**.

Fixing problems with schedulers

This section explains how to restart the **cron** and **lpsched** daemons and how to grant a user **chmodsugid** kernel authorization to use **at**(C).

cron daemon is not running

The **cron** daemon executes commands submitted with the **at**(C), **batch**(C), and **crontab**(C) commands at specified dates and times. (The **cron** daemon is started automatically by a script in the */etc/rc2.d* directory when the system is started in multi-user mode.) If **cron** is not running and you try to submit a job with one of these programs, the system displays the following message:

```
cron may not be running - call your system administrator
```

Another indicator that **cron** is not running is if previously scheduled jobs are not being executed. Use the following command to see if the **cron** daemon is running:

```
ps -ef | grep cron
```

If there is no **cron** process, then the daemon is not running. To start **cron**, simply reboot the system. During startup, the system displays a message like the following:

```
! *** cron started *** pid = 4612 Mon Dec 18 17:44:19 PST 1989
```

The jobs scheduled with **at**, **batch**, and **crontab** should now execute properly. If **at** and **batch** continue to report error messages, see section "at command fails: cannot change mode of job" later in this chapter.

lpsched print scheduler is not running

If the print service stops in the middle of a print request, or does not start any new print jobs, determine if the **/usr/lib/lpsched** daemon is running. To do this, enter the following command:

```
lpstat -r
```

(The **lpsched** daemon is started automatically by a script in the */etc/rc2.d* directory at system startup.)

If the scheduler is down, the system displays the following message:

```
scheduler is not running
```

If this is the case, use these steps to start the **lpsched** daemon:

1. Log in as the super user *root*. (A user with **lp** authorization can also start the print daemon.)
2. Invoke **sysadmsh** and select:

   Printers ⇨ Schedule ⇨ Begin

   The system displays the following message:

   ```
   Print services started
   ```

It may take a minute or two for the printer configurations, forms, and filters to be re-established before any saved print requests start printing. Any print requests that did not finish printing when the scheduler stopped are printed in their entirety when the print service restarts. The printer configurations, forms, and filters in effect when the print service stops are restored when it restarts.

at command fails: cannot change mode of job

If you do not have the kernel authorization, **chmodsugid**, the system displays the following message when you try to invoke **at**(C) or **batch**(C):

```
can't change mode of job
```

To grant **chmodsugid** authorization so that the user can use **at**, use the following procedure:

1. Log in as *root*.
2. Invoke **sysadmsh** and select:

   Accounts ⇨ User ⇨ Examine:Privileges

3. Add **chmodsugid** to the Kernel Authorization field.

To allow the changes to take effect, the user must logout and then log in again.

This is different from the situation in which the user is not *authorized* to use **at**. If a user who is not authorized tries to use **at**, the system displays:

```
at: you are not authorized to use at.  Sorry.
```

To allow a user to use **at**, see the section "Allowing or denying individuals to use at and batch" in the "Authorizing the use of job scheduling commands" appendix of this guide.

Recovering from other system failures

This section covers problems caused by system failures, such as relieving resource constraints (messages indicating file table overflow, out of streams or queues), how to map a bad track on the hard disk, how to reset the HZ value, how to tune the system if the inode table overflows, what to do when the system panics, and how to recover from an abnormal shutdown due to power failure.

File table overflow

This message indicates there are insufficient entries in the kernel's file table. This is determined by the **NFILE** parameter.

See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on adjusting kernel parameters.

Region table overflow

This message indicates that there are insufficient entries in the kernel's region table. This is determined by the **NREGION** parameter.

See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on adjusting kernel parameters.

Out of streams

This message indicates there are insufficient Streams structures available. This is determined by the **NSTREAM** parameter.

See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on adjusting kernel parameters.

Too few free pages

This message indicates there are too few pages allocated for swapping purposes. This is determined by the **MINASMEM** parameter, found under category 4 of the **configure**(ADM) menu.

See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on adjusting kernel parameters.

Out of queues

This message indicates there are insufficient Streams queues configured into the kernel. This is determined by the **NQUEUE** parameter.

See "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide for instructions on adjusting kernel parameters.

Bad HZ value

If the */etc/initscript* system file is incorrect or corrupted at any time during system operation, the system displays the following message,

```
Bad HZ Value
```

The operating system uses the **HZ** variable to represent the system interrupt clock frequency. The **HZ** value must be set in the */etc/initscript* file to 100 clock interrupts per second:

```
HZ=100
```

If the **HZ** variable is not set correctly in */etc/initscript*, or the file is corrupted or missing, the system displays the **HZ** error message. Despite the name, **HZ** has nothing to do with the power mains.

If you verify that */etc/initscript* is present and correct, this error message may indicate that your kernel is not properly serialized. Use the following procedure to re-serialize your kernel:

1. Log in as *root*.
2. Enter:

   /etc/serialize /etc/perms/rtsmd

   Use the serial number and activation key that came with your distribution.
3. Relink the kernel with the following command:

   cd /etc/conf/cf.d
   ./link_unix

4. Reboot the system by entering **init 6**. The root prompt (#) is returned, but the shutdown process begins soon afterward.
5. Press ⟨Return⟩ at the boot prompt.

The system should no longer display the **HZ** error message.

Inode table overflow

Each open file requires an inode entry in the inode table. If the inode table is too small, the system displays one of the following error messages:

```
WARNING:System V Inode table overflow
WARNING:High Sierra Inode table overflow
WARNING:MSDOS Inode table overflow
```

When the inode table overflows, the specific request is refused. Although not fatal to the system, inode table overflow may damage the operation of various spoolers, daemons, the mailer, and other important utilities. Abnormal results and missing data files are a common result.

If the system consistently displays this error message, use the following procedure to evaluate whether your system needs tuning:

1. Determine the current size of the inode table by entering:

   crash

   You see the following display:

   ```
   dumpfile= /dev/mem, namelist= /unix, outfile= stdout
   >
   ```

   At the " > " prompt, enter **var** and press ⟨Return⟩.

   A list of system variables is displayed; "v_inode" is the current size of the inode table.

2. Determine the number of active inodes by running the following command when the system is under heavy load:

   pstat -i | head -1

3. The value for NINODE (which is the same as v_inode) should be at least 10% greater than the number of active inodes at peak use time.

If the NINODE value is too small, you should increase the value and thus reconfigure the kernel to reallocate inode table entries.

For information on tuning system parameters, refer to "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide.

Mapping a bad track

During the installation procedure, the system scans the hard disk for flaws and creates a bad track table with this information. The bad tracks listed on the table are aliased to good tracks so that the operating system avoids the areas of the disk that cannot be read or written.

If your hard disk develops a bad track after the system is installed and running, an error message like the following is displayed on the console:

```
wd: ERROR on fixed disk ctlr=0 dev=0/47 block=31434 cmd=00000020
    status=00005180, sector = 62899, cylinder/head = 483/4
```

If this error occurs, use **badtrk**(ADM) to create a new bad track table so that the system can avoid the new bad track(s). The **badtrk** utility is a menu-driven utility for viewing, adding, or deleting entries in the bad track table.

WARNING If a SCSI disk displays a bad track, this indicates a serious hardware problem. Replace the disk or host adapter immediately.

To use **badtrk**, do the following:

1. Log in as *root* and enter single-user mode by typing **/etc/shutdown su**.
2. Enter **badtrk** at the system prompt.
3. Select option **2** to scan the disk, then select option **1** to scan the entire UNIX system partition.
4. Indicate whether you want to do a quick or thorough scan and, at the destructive scan prompt, enter **n**.

   NOTE Run **badtrk** in nondestructive mode to save the data on your hard disk. The thorough scan is recommended if new bad tracks have appeared.

5. When the scan is complete, the main menu reappears. Select **q** to return to the system.

The **badtrk** utility automatically enters any flaws it detects in the bad track table. For more information on the options to **badtrk**, see the manual page on **badtrk**(ADM).

Recovering from a system panic

The system "panics" when it encounters a hardware problem or kernel inconsistency that is so severe that the system cannot continue functioning. When this happens, the system displays a message on the console, and all system activity stops.

The system panic messages contain the word "PANIC" in the *severity* field, followed by a diagnostic message. For a list of the system panic messages, see the manual page **messages**(M). The system can crash without displaying a PANIC message. When this happens, the system simply refuses to process any input from the system console and all other terminals.

To recover from a system panic:

1. Copy the full PANIC message (including CPU registers), if any, from the console screen to your system log book.
2. Power-cycle the machine and press ⟨Return⟩ at the boot prompt to reboot the system.
3. At the prompt to check the root filesystem, answer **y**.
4. Bring the system up in single-user mode and run **fsck**(ADM) on those filesystems that were mounted when the system panicked. For example, use **fsck /dev/u** to check the */u* filesystem.
5. Fix the problem that caused the system to panic.

If the system does not restart, or crashes each time you start it, the operating system is corrupted and must be restored or reinstalled. To do this, follow procedures in the section "Restoring a corrupted root filesystem" earlier in this chapter.

NOTE If you cannot start the system from the boot floppy disk in the distribution set for installation, the computer has a serious hardware malfunction. Contact a hardware service representative for help.

In most cases, simply rebooting the system solves the problem. However, if your system consistently shuts down with the same PANIC message, you should fix the problem that is causing the system to panic.

For example, if the call-out table contains too few entries, the console consistently displays the following PANIC message when it crashes:

```
PANIC:Timeout table overflow
```

Fix this problem by increasing the value of the **NCALL** kernel parameter. For information on tuning system parameters, refer to "Reallocating kernel resources with configure" in the "Tuning system performance" chapter of this guide.

Recovering after a power failure

When your system goes down as the result of a power failure, the shutdown is considered "abnormal." When the system is shut down normally, the **shutdown**(ADM) program stops all daemons, kills the active processes, unmounts any mounted filesystems, tells **init** to enter single-user mode, and runs the **sync** command.

If the system goes down before this shutdown procedure completes, the following may occur:

- Filesystems may be inconsistent or corrupted, resulting in lost data.
- Ongoing work by users and other data may be lost because the buffer cache was not flushed to disk.

Because UNIX System V writes to disk every 60 seconds, the amount of data lost due to system shutdown without the **sync** command should be minimal. However, because the filesystems were not unmounted properly before the system shut down, filesystem corruption may be extensive. If the root filesystem is corrupted, the system does not function properly.

NOTE **SLEEPTIME** in */etc/default/boot* defines the number of seconds between "syncs".

When the power fails, turn the machine off. This minimizes damage to your system if the power fluctuates.

Once the power comes back on, restore your system using the following steps:

1. Turn on the computer and press ⟨Return⟩ at the boot prompt.
2. If the root filesystem is corrupted, the system prompts you whether to check the */dev/root* filesystem; enter **y** to invoke **fsck**(ADM). The system may display the following message:

   ```
   FREE INODE COUNT WRONG IN SUPERBLK
   FIX?
   ```

   This message is routine when the system is not shut down properly. Enter **y** and **fsck** fixes the problem.

   The **fsck** command automatically fixes any problems with the root filesystem when autobooting, if the */etc/default/boot* file contains the following line:

   ```
   FSCKFIX = YES
   ```

   If */etc/default/boot* does not contain this line, **fsck** prompts you for instructions on how to fix the problems that it encounters.
3. If any non-root filesystems are corrupted when the system enters multiuser mode, the system prompts you whether to clean the filesystem; enter **y**.

   The **fsck** command automatically fixes any problems with the filesystem at this time if the entry for that filesystem in the */etc/default/filesys* file reads:

   ```
   fsckflags = y
   ```

   If the filesystem line in */etc/default/filesys* does not contain this line, **fsck** prompts you to fix any problems with the corrupted filesystem.

Resolving security-related error messages

This section lists problems and error messages that you may encounter. Each problem is discussed in context, including the reason for the situation, the solution to the problem, and ways to prevent the situation from recurring. The problems described here require logging in as *root* to fix them. Because the system can potentially lock out all users (including *root*), you should be aware of the existence of the "override" terminal for *root*. If you are locked out and you removed the default override tty, you must reset the system or power cycle it, and come up in single-user (maintenance) mode to enter the system. Because shutting off the system can lead to filesystem damage, it is critical that you have an override terminal.

Account is disabled -- see Account Administrator

If you see this message, it means the account is locked for one of three reasons:

- You locked the account through the **sysadmsh** selection Accounts ⇨ User ⇨ Examine:Logins. If you want to re-establish the account, use the same selection to unlock the account.
- The password lifetime for the account elapsed. The password for the account did not change before the password lifetime was over. To re-enable the account, you can either assign a new password or use the Accounts ⇨ User ⇨ Examine:Logins selection to clear the lock (in which case the user is forced to set the password at login time). Advise users to change their passwords before the lifetime expires.
- A number of unsuccessful tries were made on the account, exceeding the threshold number you set for locking it. These tries may not have all been made on the same terminal. Before re-enabling the account, it is a good idea to determine the cause for the lock-out. It may be that the user is a poor typist, or another person is trying to lock the account and knows this is a way to do it, or a real attempt to penetrate the account is being made. You may want to adjust the threshold upward or downward, depending on the nature of the system users, the value of the data, and the accessibility of the system to outsiders.

Account is disabled but console login is allowed

This message is associated with the super user logging onto the override terminal. Under the assumption that the console device (including a serial console) is a special device and considered a physical resource worth protecting, a lock on the super user account does not prevent the super user from logging into the console. This presents a means for entering the system even when all other accounts or terminals are locked. Before continuing, use the audit trail to investigate the reasons for the lock. A lock-out caused by unsuccessful login attempts on the system console is cleared automatically, but lock-outs due to other reasons remain in effect. The console, in effect, is never locked out for the super user.

Terminal is disabled but root login is allowed

This message is associated with the super user logging onto the override terminal. Under the assumption that the console device (including a serial console) is a special device and considered a physical resource worth protecting, a lock on the super user account does not prevent the super user from logging into the console. This presents a means for entering the system even when all other accounts or terminals are locked. Before continuing, use the audit trail to investigate the reasons for the lock. A lock-out caused by unsuccessful login attempts on the system console is cleared automatically, but lock-outs due to other reasons remain in effect. The console, in effect, is never locked out for the super user.

Audit: file system is getting full

The audit subsystem may occasionally display this warning when the audit filesystem reaches a certain threshold. This warning message indicates that space is low on a particular device. If additional directories were specified to the subsystem, it automatically switches when the filesystem has reached the threshold value for remaining free space. Otherwise, the administrator must intervene to make more space available. If not, auditing is terminated when the threshold value is reached. The audit daemon program **auditd** may terminate for the same reason. If unable to write a compaction file because of insufficient space or an I/O error, **auditd** terminates. Use the **sysadmsh** System ⇨ Audit ⇨ Disable selection to terminate auditing if it has not already been done. Analyze the source of the problem and solve it before re-enabling auditing.

Authentication database contains an inconsistency

This message is displayed while running one of the programs associated with the TCB or a protected subsystem. The Authentication database integrity is in question. The Authentication database is a composite of the Protected Password database, the Terminal Control database, the File Control database, the Command Control database, the Protected Subsystem database, and the System Defaults file, and the message applies to all of these. Either a data entry is not present when expected, or the items within an entry are not correct. This message is intentionally vague. The invoking user is alerted to the problem but not given enough information about the cause to allow them to exploit an integrity problem within the security perimeter. The real reason for the problem may be found in the audit trail if Database Events were enabled for the user that generated the message. The Accounts ➪ Check ➪ Databases selection should help you determine the problem.

Can't rewrite terminal control entry for tty

The most likely reason for this problem is that the device tty entry in the */etc/auth/system/ttys* file is corrupted. If a multiscreen entry in the */etc/auth/system/ttys* file is corrupted, the multiscreens listed below the corrupted entry are inaccessible. For example, if the entry for **tty03** is corrupted, when you press ⟨Alt⟩⟨F4⟩ or ⟨Alt⟩⟨F5⟩, the system displays a blank screen.

To solve this problem:

1. Log in on the override tty.
2. If the */etc/auth/system/ttys* file does not contain an entry for **tty01,** add the following line below the **console** entry:

   ```
   tty01:t_devname=tty01:chkent:
   ```

   You can now enter multiuser mode and access the multiscreens of the console.
3. If the */etc/auth/system/ttys* file appears intact, you should check and see if the */etc/group* file was corrupted or removed. If so, restore the file from backups.

Authentication error; see Account Administrator

The most likely reason for this problem is that the device tty entry in the */etc/auth/system/ttys* file is corrupted. If a multiscreen entry in the */etc/auth/system/ttys* file is corrupted, the multiscreens listed below the corrupted entry are inaccessible. For example, if the entry for **tty03** is corrupted, when you press ⟨Alt⟩⟨F4⟩ or ⟨Alt⟩⟨F5⟩, the system displays a blank screen.

To solve this problem:

1. Log in on the override tty.
2. If the */etc/auth/system/ttys* file does not contain an entry for **tty01**, add the following line below the **console** entry:

   ```
   tty01:t_devname=tty01:chkent:
   ```

   You can now enter multiuser mode and access the multiscreens of the console.
3. If the */etc/auth/system/ttys* file appears intact, you should check and see if the */etc/group* file was corrupted or removed. If so, restore the file from backups.

Cannot obtain database information on this terminal

This message indicates a problem with the */etc/auth/system/ttys* file. This message can be generated sporadically on a system with a large number of users logging on and off (as on a BBS). However, if the message is displayed consistently on all terminals, it is necessary to check the database files. Running the **tcbck**(ADM) utility should fix any problems with the *ttys* file. Under normal conditions, **tcbck** is run automatically whenever the system boots.

Login incorrect

The user entered an incorrect login name or login/dial-up password. If this happens repeatedly, you may need to alter the password to permit the user to log in again.

NOTE There are several messages associated with logins. In some cases, there are also additional explanatory messages generated by **sysadmsh**. The **sysadmsh** messages are shown in normal rather than screen font to avoid confusing them with **login** messages.

When a user (including *root*) cannot log in and the cause is not related to a locked terminal or account, it is possible that trusted database files are missing or corrupted. If the user sees the "Login incorrect" message and they are certain that their password is correct, it is possible that the */etc/passwd* and/or Protected Password database is corrupted. You must use the Accounts ➪ Examine selection to determine the real problem.

When you enter the username, the account information is checked and more descriptive error messages are generated by **sysadmsh**. These **sysadmsh** messages and the necessary remedies are as follows:

User exists in */etc/passwd* but not in Protected Password Database
: The database entry located in */tcb/files/auth/*[*a-z*]/***username*** was deleted or corrupted. If you find that this is true for a number of users, you should restore the files from backups. For an individual user, the easiest way to repair this is to select the Identity information and replace any empty fields and assign a new password.

User exists in Protected Password Database but not in */etc/passwd*
: Somehow */etc/passwd* was corrupted or someone with root powers made faulty edits to the file. You should try to select the Identity information and replace missing fields. If this does not work, you must log in as *root* and restore your */etc/passwd* file from backups.

login: resource Authorization name file could not be allocated due to: cannot open;

The */etc/auth/system/authorize* file was corrupted or removed. Log in on the override tty and restore the file from backups.

Terminal is disabled -- see Account Administrator

The terminal is locked to all users. This is similar to account locking. Either an authentication administrator locked the account with the **sysadmsh** or a number of incorrect login tries (to one or more accounts) passed the threshold for that terminal. In both cases, determine what happened and then use the **sysadmsh** to reset the lock.

Bad login user id

The */etc/auth/system/authorize* file was corrupted or removed. Log in on the override tty and restore the file from backups.

useshell: File Control database inconsistency

This means there is a problem with the permissions or ownership of a database file located in */usr/lib/mkuser.* You should run the **fixmog**(ADM) utility as follows:

```
fixmog -i
```

The **-i** option checks files interactively, asking you to to confirm each change before it is done. There may be additional problems reported by **fixmog** that should be examined via an **integrity** report; see "System file integrity checking: integrity(ADM)" in the "Maintaining system security" chapter of this guide.

useshell: Script path /usr/lib/mkuser/mkuser.init may be compromised...

This means there is a problem with the permissions or ownership of a database file located in */usr/lib/mkuser.* You should run the **fixmog**(ADM) utility as described for the previous item.

You do not have authorization to run...

The command is part of a protected subsystem. For that subsystem, the authentication administrator has not provided you with the kernel authorization needed to run this command and/or related commands. The authentication administrator uses the Accounts ➪ User ➪ Examine:Privileges selection to grant or deny such authorizations.

Troubleshooting your tape drive

This section covers some common problems that you may have with your cartridge, floppy, or SCSI tape drive on your system. Note that these problems generally occur immediately after using **mkdev tape** to install your tape drive.

For more information about installing and configuring tape drives, see the "Using floppy disks and tape drives" chapter in this guide.

Bad octal digit

After you run **mkdev tape** to install a cartridge tape drive, the system displays the following message:

```
/etc/conf/pack.d/ct/space.c line 46 bad octal digit
```

To fix this problem, verify that the base address for the tape controller is entered with a leading "0x" rather than a trailing "H". Run **mkdev tape** again to modify the cartridge tape parameters.

Tape not recognized at bootup

If your system does not display a message like the following at boot time, the system does not recognize the tape controller card:

```
%tape          0x338-0x33C    05         1     type=W
```

Instead of this hardware recognition message, the system displays a message like the following:

```
ct: ERROR: Tape controller (type=W) not found
```

Cartridge tape

For a cartridge tape drive, check the following:

1. Verify that the tape controller card is physically configured to the base address that you gave when you ran **mkdev tape**. To check this, run **hwconfig**(C). If this value is incorrect, either physically reconfigure these settings on the tape controller so that it agrees with the values that you indicated when you ran **mkdev tape**, or run **mkdev tape** again and change the parameters.
2. Use **hwconfig** to verify that the DMA (Direct Memory Access) Channel, base address, and interrupt vector do not conflict with any other hardware device on your system. If one or more of these values conflicts, run **mkdev tape** and modify the settings.

Table 20.1 lists the interrupt (IRQ) vectors that are commonly used by ISA system hardware.

Table 20-1 Interrupt vector usage (ISA)

Interrupt	Device
0	clock
1	console
2	available (often entered as **9**)
3	COM2 (**tty2a**)
4	COM1 (**tty1a**)
5	**lp2** (alternate parallel port)
6	floppy disk controller
7	**lp0** or **lp1** (main parallel port)

NOTE Do not use interrupts 0, 1, and 6.

For more information about configuring interrupts on your cartridge tape drive, see the section "Using tape drives" in "Using floppy disks and tape drives" in this guide.

3. Verify that the tape controller is seated properly on the motherboard and, if necessary, insert the controller in a different slot.
4. Some tape drives (particularly external drives) require that the drive be attached to the tape controller and powered on at boot time.
5. Verify that the cartridge tape drive is supported. See Appendix A, "Compatible hardware," of the *Release Notes* for a list of the compatible tape drives.
6. If your system still does not recognize the tape controller card at boot time, the tape controller is broken. See the documentation that came with your hardware for more information.

Irwin and QIC-40/80

For an Irwin (mini cartridge) or QIC-40/80 tape drive, verify that the jumper is set for the correct floppy position on the back of the floppy tape drive.

SCSI tape

For a SCSI tape drive, run **mkdev tape** to check the following:

1. Verify that the ID number for the controller of the device is correct. The ID number is determined by the jumper settings on the controller. The valid range is 0-7.
2. Make sure that the host adapter number is correct. The first SCSI host adapter is 0; the second is 1.

3. Check that the LUN (Logical Unit Number) is correct. In most cases, the controller is embedded in the same physical unit as the device and supports one device with LUN 0. If the controller is not embedded, it supports up to eight devices. If this is the case, the LUN is determined by the jumper settings on each device. The valid range is 0-7.
4. Verify that the tape drive is supported. See Appendix A, "Compatible hardware," of the *Release Notes* for a list of the compatible tape drives.

Tape commands hang

If the tape drive hangs when you test it with the following command:

tape reset

Check the following:

1. Verify that the cartridge tape controller card is physically configured for the DMA value that you gave when you ran **mkdev tape**. To fix this, either physically reconfigure the tape controller DMA setting to agree with the address that you gave with **mkdev tape**, or run **mkdev tape** again and specify the DMA value on the tape controller. Do the same for the interrupt vector. Remember that hardware interrupt 2 is entered as 25.
2. Verify that the cable between the tape controller and the tape drive is properly attached.
3. If the **tape reset** command still hangs, either the tape controller, the tape drive, or the cable is broken. See the documentation that came with your hardware for more information.

Cannot open /dev/rct0

If a **tar**(C) command or other commands which access the tape drive fail to write the contents of a directory to a tape and the system displays the following message:

```
Cannot open /dev/rct0
```

Check the following:

1. Use **hwconfig** to verify that the interrupt vector for a cartridge tape drive does not conflict with any other device on your system. See the section "Tape not recognized at bootup" earlier in this chapter for a list of the available interrupt vectors. To change the interrupt vector, run **mkdev tape** and indicate a new interrupt vector value.
2. Verify that the actual physical interrupt (IRQ) setting on a cartridge tape controller card agrees with the interrupt vector that you specified with **mkdev tape**.

3. Make sure that the proper device for the tape drive is located in the */dev* directory. Enter:

   l /dev/rct0

   The output should look like the following:

   ```
   crw-rw-rw   1 root  root  10, 0 Feb 14 12:00 rct0
   ```

   (The major device number (10) may differ.) If the listing of */dev* does not contain a line similar to this, run **mkdev tape** again to create the device.
4. Verify that the cable between the tape controller card and the tape drive is connected correctly.
5. If the **tar** command still does not work, either the tape controller, the tape drive, or the cable is broken. See the documentation that came with your hardware for more information.

Solving terminal problems

This section explains how to solve the following problems with terminals:

- Terminal is completely non-functional
- Terminal is hung
- Terminal display is scrambled
- Terminal is locked
- Terminal does not echo

Restoring a non-functional terminal

A completely non-functional terminal displays no login prompt and does not respond to keyboard input. This situation is usually caused by hardware failure or configuration problems. Follow these steps to check a non-functioning terminal:

1. Check the brightness control on the terminal.
2. Check the power and communication connections at the terminal and computer.
3. If applicable, enter set-up mode on the terminal and verify the terminal configuration settings. The settings should include 9600 baud, 8 data bits, 1 stop bit, and no parity.
4. Enable the port to which the terminal is connected. For example, to enable tty007, use the following command:

   enable tty007

5. Verify that there is a **getty** process associated with the terminal port. For example, enter:

   ```
   ps -t tty007
   ```

6. Test the hardware communications by disabling the port and redirecting output to the non-functional terminal: for example, to test tty007, use the following commands:

   ```
   disable /dev/tty007
   echo hello > /dev /tty007
   ```

If these steps do not restore the non-functional terminal, check the documentation that came with the terminal hardware for troubleshooting suggestions.

Fixing a hung terminal

A terminal is considered "hung" if the previous work session is still visible on the display, but it does not respond to keyboard input. Use the following steps to fix a hung terminal:

1. Wait a minimum of 60 seconds before trying to resurrect the terminal. (If the system is busy, the terminal may not respond immediately to keystrokes because the system response time has increased.)
2. Press ⟨Ctrl⟩q to re-enable transmission in case the ⟨Ctrl⟩s (transmit off) signal was inadvertently pressed.
3. Check to see that all power cords, keyboard cords, and communications cables are connected.
4. Reset the terminal hardware by recycling power to the terminal and then reinitialize it by running **tset**(C) with no arguments.
5. Verify the terminal set-up mode configuration settings (if available) as described in step 3 of the previous section.
6. Test the hardware communications by redirecting output from an operating terminal to the locked one as described in the step 6 of previous section.
7. Check the processes that are running on the locked terminal port with the following command:

   ```
   ps -t ttyn
   ```

   Stop the process that the user was running when the terminal hung using **kill**(C). For information on how to do this, see the section "Stopping a runaway process" earlier in this chapter. If the program does not die, you must reboot the system to stop the process. See the section "Removing an unkillable process" earlier in this chapter.

8. Determine whether the current line characteristic parameters are correct. For example, use the following command to display these values for tty007:

   ```
   stty -a < /dev/tty007
   ```

   You can also compare the **stty** settings with those of a working terminal.
9. Reset the serial line characteristics with the following command:

   ```
   〈Ctrl〉j stty sane 〈Ctrl〉j
   ```

   If you cannot enter the command on the terminal, you can redirect the **stty** command from another terminal as follows:

   ```
   stty sane < /dev/tty007
   ```

If the **ps -t** command shows only a **getty** program, the terminal should display a login prompt. If it does not, check the terminal hardware again.

Fixing a scrambled terminal display

A scrambled terminal responds to keyboard input but the display is incorrect. Follow these steps to fix a scrambled terminal:

1. Check the terminal type (**TERM**) for the user with the **env** command. If the terminal type is incorrect, reset it. For example, to set the terminal type to **wyse60**, enter:

   ```
   TERM=wy60
   ```

   After resetting the terminal type, reinitialize the terminal by entering **tset** with no arguments.
2. Reset serial line characteristics with the following command:

   ```
   〈Ctrl〉j stty sane 〈Ctrl〉j
   ```

Unlocking a locked terminal

If a terminal has been locked by the system administrator to prevent logins on that terminal, or if the system locked the terminal because a user exceeded the number of unsuccessful logins attempts, the following message is displayed on that terminal:

```
Terminal is disabled -- see Account Administrator
```

To unlock the terminal, invoke **sysadmsh** and select:

Accounts ➪ Terminals ➪ Unlock

For more information, see "Locking or unlocking a terminal" in the "Administering user accounts" chapter of this guide.

Restoring a non-echoing terminal

A non-echoing terminal is a terminal that responds to keyboard input but does not display the characters entered at the keyboard. (This is different from a *locked* or non-functional terminal that does not respond to input at all; see "Unlocking a locked terminal" or "Restoring a non-functional terminal" earlier in this chapter for a solution to these problems.)

Sometimes, when a program stops prematurely as a result of an error, or when the user presses the ⟨Break⟩ key, the terminal stops echoing. To restore the terminal to normal operation, enter the following:

⟨Ctrl⟩**j stty sane** ⟨Ctrl⟩**j**

Enter this command accurately because the terminal does not display what you enter at the keyboard.

The terminal should now display keyboard input. If it does not, follow the steps outlined in the section, "Restoring a non-functional terminal" earlier in this chapter.

Solving mouse problems

The error messages in this section are associated with mouse problems. If you only recently installed the mouse, make certain you checked all the points covered in the "Adding mice and other graphic input devices" chapter of this guide.

usemouse: no mouse available(5)

Try executing the **tty**(C) command on the multiscreen that is failing. If the **tty** command is executed on the first multiscreen, it will normally return "/dev/tty01". If **tty** returns "/dev/syscon" on the failing multiscreen then the system was shutdown on that multiscreen using the **init s** or **init S** command. Shutting down the system with **init s** or **init S** will cause the system to enter single-user mode. As documented in the **init**(M) manual page, this will also cause the system console device (*/dev/syscon*) to remain linked to the tty from which the last **init S** was invoked.

To correct this problem, and allow the mouse to work, take the following steps:

1. Log in as *root* and run **mkdev mouse**.
2. Select option **4**, "Associate a terminal with an existing mouse."

3. Enter **syscon** as the terminal device.
4. Enter **exit** to exit from **mkdev mouse**.

This adds */dev/syscon* to the list of terminals with which the mouse can be used.

usemouse: can't open slave (errno 13)

This message means that **usemouse** cannot open the slave side of the pseudo-tty that is necessary to communicate with the system.

Check the permissions on the "master" pseudo-ttys (devices beginning with */dev/ptyp*) and "slave" pseudo-ttys (devices beginning with */dev/ttyp*). They should be 666 mode or "rw-rw-rw-". If they are not correct, change them using the **chmod**(C) command. As a general rule, if a problem is related to permissions, running as a normal user gives the problem, but running as root does not. In addition, check the entries in the file */usr/lib/event/ttys* which indicate that a mouse is a legitimate "event source" for the tty you are on. For example:

```
tty01 msbusmouse
```

This entry indicates that you are attempting to use a Microsoft busmouse on the first console screen.

Troubleshooting network connections

This sections discusses common problem situations that you may experience with UUCP and covers the following topics:

- How to check for a faulty ACU/modem
- Error messages that **cu** displays
- Common errors when running **uucp**
- How to check the status of a **uucp** request
- Using **uutry** to debug UUCP communications
- Common messages that appear in the UUCP log and status files
- How to check permissions settings on UUCP files
- Checking that your sitename is unique
- What to check if UUCP is very slow
- What to do when **uucp** works, but **uux** does not
- A description of the UUCP troubleshooting utilities, **uucheck, uulog, uuname, uustat,** and **uutry**.

Before troubleshooting the UUCP system, make sure that the physical connection works. If you are connecting to UUCP with a modem, verify that the modem is installed and configured correctly. If you are using a direct line, make sure that the computers are connected correctly. Use the instructions in the sections "Connecting two local systems using a direct wire" in the "Building a remote network with UUCP" chapter and the "Using modems" chapter of this guide.

NOTE **cu** must work in order for UUCP to work.

Check for faulty ACU/Modem

The following are two methods for checking whether the ACU (Automatic Call Unit) or modems are working correctly:

- To display a list of queued requests, the time of the last request attempt, and the contact status, enter:

  uustat -q

- To use a specific line and print debugging information during the contact attempt, enter:

  cu -x9 -l ***tynn phone***

  NOTE To protect the modem from interference from unqualified users, this command is only permitted for those who have write access to the *Devices* file.

For information on solving common modem problems, see the section "Troubleshooting your modem" earlier in this chapter.

Errors when testing the connection with cu

This section describes messages that the system displays when testing the connection with the **cu** command fails.

Connect failed: CANNOT ACCESS DEVICE

If the connection fails and the system displays the CANNOT ACCESS DEVICE message, check the permissions on the device file. For example, to check the device file for **tty1a**, enter:

l /dev/tty1a

The ownership and permissions settings should look like the following:

```
crw--w--w- 1 uucp  uucp     5, 0 Feb 14 12:00 /dev/tty1a
```

If, instead, the ownership and permissions for the device file look like this:

```
crw------- 1 bin   terminal 5, 0 Feb 14 12:00 /dev/tty1a
```

verify that the line for the port in the */etc/inittab* looks similar to the following:

```
t1A:23:off:/etc/getty -t60 tty1A 3
```

If it does not, edit */etc/inittab* and */etc/conf/init.d/sio* and change the lines for the appropriate port.

NOTE Because each time the kernel is relinked, a new *inittab* file is created, to retain the changes in *inittab*, you must also edit */etc/conf/init.d/sio.*

To make your changes to *inittab* effective immediately, enter:

telinit Q

Connect failed: SYSTEM NOT IN systems FILE

If the */usr/lib/uucp/Systems* file on your computer does not contain an entry for the system that you are trying to access, **cu** displays the SYSTEM NOT IN Systems FILE error message.

To display a list of all the systems that your system is connected to, enter

uuname

The system may display this error message if you used the **-l** option to specify a serial port and you entered the wrong line number. Verify that the line is configured properly.

Connect failed, NO DEVICES AVAILABLE

If **cu** fails to connect and displays the NO DEVICES AVAILABLE message, check to see that the */usr/lib/uucp/Devices* file is set up correctly.

Verify that the line that corresponds to the device that you are using is uncommented. For example, the entry in *Devices* for a direct line using **tty1a** at 9600 baud looks like this:

```
Direct tty1a - 9600 direct
```

If the *Device* file looks correct, the remote line may be busy. Try again later.

Connected, but no login prompt

If **cu** displays the **Connected** message, but not the login prompt, the line for the remote system may be busy. To exit **cu**, enter **~.** and press ⟨Return⟩.

If everything appears to be working on your system but the login prompt for the remote machine does not appear, check with the remote system administrator to verify that the **getty** on the remote system is set up with the same communications parameters that you are using.

Connected, but screen displays garbage

If you connect to the remote system, but your screen displays garbage, the connection may be bad. Exit **cu** by entering `~.` and try again later.

Another possibility for this situation is that the communications settings on your system are different from the settings on the remote system. Check with the system administrator on the remote system.

UUCP failed messages

Errors that UUCP displays immediately after you enter the **uucp** command are generally syntax or permissions errors, and include either of the following messages:

```
uucp failed completely
uucp failed partially
```

The following are some common problems and the error messages that the system displays:

- UUCP displays the following message when the permissions are set so that **uucp** cannot access the source file:

  ```
  can't read file (filename)
  uucp failed partially
  ```

  Change the permissions on the source file to allow all users read permissions.

- The following message is displayed when you do not enter both a ***source*** and ***destination*** with the **uucp** command:

  ```
  usage uucp from ... to
  uucp failed completely
  ```

- If you enter a sitename that is not in the */usr/lib/uucp/Systems* file, UUCP displays:

  ```
  bad system name: sitename
  uucp failed completely
  ```

  Use **uuname** to display a list of the sites to which your system is connected.

Checking the status of a uucp request

If the system does not display an error message immediately after entering the **uucp** command, and the system prompt appears, the **uucp** request is queued. Use the **uustat** and **uulog** utilities to check the status of your **uucp** job.

Use the **uustat** command to display the status of the currently queued **uucp** requests or display the status of connections to other systems. For example, to display a list of all the queued jobs for the user *robertm*, enter:

uustat -u robertm

The **uustat** utility displays user status information in the following format:

```
couscousN266 01/26-15:43 S couscous robertm rmail edwarda
```

To list the status of the accessibility of all the remote machines, use the following command:

uustat -m

The following example shows the output from the **uustat** command:

```
scooter 01/26-15:43  CAN'T ACCESS DEVICE
disco   01/27-12:01  SUCCESSFUL
obie    01/25-05:12  CALL FAILED, JOB DELETED
```

For more information on the options to **uustat**, see the **uustat**(C) manual page.

The **uulog** command displays the status messages (most recent last) that the UUCP programs write to the files *uucp/sys, uux/sys, uucico/sys,* and *uuxqt/sys,* in the */usr/spool/uucp/.Log* directory (*sys* is the name of the system that **uucp** is trying to access).

For example, to display status information about file transfers involving the system *scooter*, enter the following:

uulog -s scooter

The **uulog** utility displays information about the system in the following format:

```
uucp scooter (01/26-15:43:00, 304, 5) COPY (SUCCEEDED)
```

NOTE The **uulog** command displays all the status messages that **uucp** logs in the *.Log* files; the messages are not necessarily error messages.

Debugging UUCP communications

If you are unable to contact a particular machine, use the following procedure to debug the communications to that machine with **uucp** and **uutry**.

1. Log in as *root*.
2. Enter the following commands, pressing ⟨Return⟩ after each line:

   uucp -r ***testfile machine*****\\!/usr/tmp/*testfile***
   /usr/lib/uucp/uutry -r ***machine***

   where ***testfile*** is a file that you want to transfer and ***machine*** is the sitename of the machine that you are trying to contact. The **-r** option overrides the retry time specified in */usr/lib/uucp/Systems*.

   NOTE If you are using **csh**, you must escape the **!** (\\!).

3. When you see one of the following messages, press the ⟨Del⟩ key:

```
Conversation Complete: Status SUCCEEDED
Conversation Complete: Status FAILED
```

If your system displays the first message, check the remote system, ***machine*** for the file */usr/tmp/testfile*. If the file exists and is not empty, the UUCP connection functions. If the system displays the second message, the connection transfer failed.

If the UUCP system is set up correctly, the system attempts to transfer the *testfile* file to the remote system, ***machine***. The **-r** option to **uucp** queues the file without starting the transfer. This allows **uutry** to invoke **uucico** with debugging.

NOTE The **uutry** command displays more debugging information if you run it as *root*.

The **uutry** command saves the debugging output to the file */tmp/machine* on the local machine and prints the debugging output to your terminal using **tail -f**. Copy the output from */tmp/machine* if you wish to save it. You can change the debugging level from the default level (five) using the **-x** option.

If the system displays the "Status FAILED" message, you can look through this file for status and error messages.

See the section "UUCP log and status file messages," for an explanation of some common UUCP messages.

If you still cannot solve the problem, you may need to call support personnel. Save the debugging output for diagnosing the problem.

Alarms in UUCP audit output, data is not transferring

A problem can sometimes occur once UUCP has begun sending data, in that part way through the transfer, alarms are generated and UUCP attempts to resend the packets. There are a couple of possible reasons for this problem.

First, check that **mapchan** is not being used on the line that **uucico** is using. If it is, enter:

```
mapchan -n ttyxx
```

to turn mapping off for that port. If **mapchan** is not running on that port, there is probably some kind of line noise or modem problem which is causing the data being sent to be corrupted.

Alternatively, the problem may occur if you are using UUCP with MNP 5 or some other error checking protocol that is specific to the modems. UUCP expects a certain packet size and then a checksum on the packet. If it does not receive a full packet in a certain amount of time, a timeout occurs which generates an alarm. Error checking on a modem also sends a certain packet size with a checksum. The problem is that the sending modem waits to get a certain amount of data before it sends the packet, which can cause problems with UUCP because it is also waiting for a packet. The solution is not to use error checking on the modem when using UUCP because UUCP has its own error checking.

UUCP log and status file messages

UUCP keeps track of activity in log and status files in the */usr/spool/uucp* directory. The UUCP programs **uucp, uux** , **uucico** , and **uuxqt** write status information for each system to files in the */usr/spool/uucp/.Log* directories. The**uucico** utility writes messages to the files in the */usr/spool/uucp/.Status* directory. These messages describe the status of the transfer request, such as **uucico** failures, completions, and the time until the next allowable call for each remote system with which you communicate.

The following sections describe common error messages that UUCP programs write to these files.

DEVICE LOCKED

The **uucico** creates a lock file named *LCK..**sitename*** for the remote system in the */usr/spool/locks* directory. If a file for the system that you are trying to call exists, **uucico** thinks that the device is in use.

To use the device, enter the following command on the remote system:

```
rm /usr/spool/LCK*
```

NOTE If this does not solve the problem, the modem could be asserting CD improperly.

LOGIN FAILED

If this error message appears, one of several things may be wrong:

1. Check the */usr/lib/uucp/Systems* file on the local machine to verify that the information in this file, particularly the chat script, is current. Some information that may be out-of-date is the phone number, login, and password.
2. Verify that the modem setup on both ends is correct. See the section, "Troubleshooting your modem," earlier in this chapter for more information.
3. Check the phone connection; try a different phone line.

After each step, invoke the **uutry** command.

NO DEVICES AVAILABLE

If the system displays this message, one of several things may be wrong:

1. The system thinks that the modem is in use because there is a lock file for it. Enter the following from the local machine:

   ```
   rm /usr/spool/LCK*
   ```

2. There may be no valid device for the calling system to use. Verify that the device named in the */usr/lib/uucp/Systems* file corresponds to the entry in */usr/lib/uucp/Devices*.
3. If the message persists, reboot the system. Enter the **rm** command again and then invoke the **uutry** command again.
4. If the message persists after rebooting the system, the modem may be configured incorrectly. Verify that you can use **cu** to call out. For more information, see the section "Troubleshooting your modem" earlier in this chapter.

REMOTE DOES NOT KNOW ME

If the system displays this message, the */usr/lib/uucp/Systems* file on the machine that you are trying to call does not contain an entry for your machine. Check the *Systems* file on the remote machine.

REMOTE HAS a LCK FILE FOR ME

If your system displays this message, either the remote system is trying to call your system or there is a lock file for your system on the remote system.

1. Remove the lock file from the remote system by entering:

   rm /usr/spool/LCK*

2. Invoke the **uutry** command again.

SYSTEM NOT IN systems FILE

If the */usr/lib/uucp/Systems* file on your system does not contain an entry for the system that you are trying to access, this error message is displayed. Verify that the system name is in the *Systems* file using the **uuname** command.

RETRY TIME NOT REACHED

When UUCP requests fail, retries are not executed immediately. After an attempt to contact a remote system, a status file remains in */usr/spool/uucp/.Status/**nodename*** (***nodename*** is the name of the remote system that you are trying to reach). This file contains information about the last request and does not allow another request until the minimum retry period (specified in the */usr/lib/uucp/Systems* file) is reached. If you try to use UUCP again, the system displays an error message like the following:

```
RETRY TIME NOT REACHED
```

To enable another call attempt immediately, remove the status file for the remote system from the */usr/spool/uucp/.Status* directory:

rm /usr/spool/uucp/.Status/*nodename*

CANNOT ACCESS FILE

If the system displays this message, it cannot access the calling device port. Check the permissions and ownership on the device file:

l /dev/tty

Where ***xx*** is the tty number. The ownership and permissions settings should look like the following:

```
crw-r--r--  1 uucp  uucp  5,  0 Feb  14 12:00 /dev/ttyxx
```

WRONG TIME TO CALL

The */usr/lib/uucp/Systems* file may restrict outgoing calls at this time.

Checking UUCP files permissions settings

UUCP does not work correctly if it cannot read or execute its files. Because virtually all of the UUCP files are writable only by the super user, and many of them are also readable and executable only by *root* and **uucp**, you should log in as *root* to install and modify the UUCP system. When you have finished, verify that all of the UUCP files are owned by **uucp** and not *root*.

To check the permissions of the UUCP files, use the following commands:

```
cd /
fixperm -n -v -dUUCP /etc/perms/*
```

This **fixperm** command displays a list of any UUCP files with incorrect permissions.

NOTE The *Systems* and *Permissions* files contain unencrypted passwords, and therefore should be readable only by **uucp** (and *root*).

Verifying that sitename is unique

Make sure that the first seven characters of your sitename name are unique. If your machine is connected directly to a machine with the same sitename, UUCP does not allow communication with that machine. If both your machine and another machine with the same name are not directly connected, but are on the same UUCP network, electronic mail may go to the wrong machine.

To check the sitename, enter:

```
cat /etc/systemid
```

To change the sitename, either edit the */etc/systemid* file manually, or invoke **uuinstall** and select option **1** to display or change your sitename.

UUCP is abnormally slow

Because UUCP is a batch network, requests are not executed immediately. Therefore, when users report that UUCP does not work, the problem may be that the system is slow. If the system is slower than usual, check for the following problems:

1. Check the spool directories, */usr/spool/uucppublic* and */usr/spool/uucp* for an overload of old work files and remove them.
2. Use **ps**(C) to verify that there are not too many **uucico** processes going at once. The limit to the number of communications processes is specified in */usr/lib/uucp/Maxuuscheds* and */usr/lib/uucp/Maxuuxqts*. Check to see that the limit is not too high; the default is 1.

3. Make sure that the filesystem that contains the spool directory is not out of space. For more information, see the section "Out of space on filesystem" under "Fixing filesystem problems" earlier in this chapter.
4. The */usr/lib/uucp/uudemon.clean* script performs general cleanup by removing old files that are trying unsuccessfully to execute. To change the frequency of the cleanings, edit the **find** statement in the *uudemon.clean* script.

uucp works, but uux does not

If you can use **uucp** to transfer files between two systems, but you cannot use **uux**, there is a problem with the */usr/lib/uucp/Permissions* file. When you use the **uucp** utility, the remote system requires only the ***LOGNAME*** entry in *Permissions;* **uux** also requires the ***MACHINE*** entry.

To fix this problem, add the ***MACHINE*** entry, with the name of the remote system, to *Permissions.* For example, if your local machine, *goanna* is set up to call *obie,* the entry for *goanna* in the *Permissions* file on *obie* should look like this:

```
LOGNAME=uugoanna MACHINE=goanna \
        COMMANDS=ALL \
        READ=/ \
        WRITE=/ \
        SENDFILES=yes REQUEST=yes
```

NOTE The permissions granted in the example above are very liberal and should only be used in closely coupled systems where there is no security risk.

UUCP troubleshooting utilities

Table 20.2 lists several commands you can use to check for basic communications information.

Table 20-2 UUCP troubleshooting tools

Command	Description
uucheck	Allows you to check for the presence of files and directories required by **uucp**. This command also checks the *Permissions* file for obvious errors.
uulog	Displays the contents of the log directories for specific hosts.
uuname	Lists the machines that you are set up to contact.
uustat	Display the status of the currently queued **uucp** requests or connections to other systems.
uutry	Invokes **uucico** with debugging, saves the information to the file */tmp/machine,* and directs the last 10 lines of the output to the terminal. The **-x** option changes the debugging level (default is level 5).

Appendix A
Customizing system startup

When your system is switched on and booted, certain aspects of system operation are set up. The system reads initialization files at startup, when changing run levels, and whenever a user logs in. By modifying these files, you can adapt system startup.

The system initialization files contain commands or data that:

- set initial run levels
- set the system clock
- enable terminals
- start programs
- check and mount specified filesystems
- clean up temporary directories
- set home directories and terminal types for users
- display system messages

The files discussed here are */etc/inittab,* the scripts in the */etc/rc2.d* directory, *.profile*, *.login,* and */etc/motd.*

The system administrator can modify the startup files to create any initial system and user environment. For example, by adding or changing entries in the *inittab* file, specific terminals can be enabled (or disabled) when the system enters or leaves a particular run level. By changing a script in the */etc/rc2.d* directory, process accounting can be started automatically at system startup. The administrator can also customize a specific user's environment by modifying the *.profile* or *.login* file in their home directory.

The initialization files are ordinary text files and can be modified using a text editor such as **vi**(C) (see the *User's Guide*). Note, however, that entries in the */etc/inittab* file must follow a specific format described in the **inittab**(F) manual page. (For more information on **init** run levels, refer to the **init**(M) manual page.) The scripts in */etc/rc2.d* and the *.profile* and *.login* files contain commands and comments. These are in the command file format described in the chapter of the *User's Guide* entitled "The shell."

Changing the /etc/inittab file

The **/etc/init**(M) program starts during the last phase of kernel initialization and has a process **id** (PID) of " 1 ". The **init** process starts all other processes. The */etc/inittab* file contains instructions for **init**. The **init** program reads the *inittab* file under three circumstances: at boot time, when an **init**-started process completes, and when the system administrator executes either the **/etc/init** or **/bin/telinit** command with a run-level argument. The arguments passed to **init** allow you to change the system run level or force **init** to examine the *inittab* file without changing the run level. When the system changes run levels, **init** scans *inittab* for instructions that apply to the new state.

NOTE When you modify *inittab,* the change is only temporary because each time the kernel is relinked, a new *inittab* file is created. To change the initialization procedure permanently, you must also modify the source from which the *inittab* file is recreated. To add a new entry, append it to the */etc/conf/cf.d/init.base* file. To modify an entry, locate and edit the existing entry in */etc/conf/cf.d/init.base* or in one of the other component files in the */etc/conf/init.d* directory.

The *inittab* file is made up of entries that contain four fields separated by colons:

label*:*run_level*:*action*:*process

Table A.1 describes the fields in *inittab*.

Table A-1 inittab fields

Field	Description
label	a unique identification label of up to four characters
run_level	the init level at which the entry is executed
action	a keyword indicating the action that init is to take on the process
process	the process init executes upon entering the specified run level

If there is more than one run-level specified for an *inittab* entry, the levels appear in the second field without separators. If the run level field is empty, the entry is executed in all numeric run levels (0-6). When the run level changes, any process that does not have an entry for the new run level receives a warning signal (15). If the process does not terminate after 5 seconds, it receives a kill signal (9). The current state of the **init** process determines how it executes the *inittab* entry.

When the **init** program is initially invoked, it scans *inittab* for entries that contain the action keywords described in Table A.2. These entries are executed only during **init**'s boot-time read of *inittab*.

Table A-2 inittab single-user keywords

Keyword	Description
boot	starts the process and continues to the next entry without waiting for the process to complete. When the process dies, **init** does not restart the process.
bootwait	starts the process once and waits for it to terminate before going on to the next *inittab* entry.
initdefault	determines which **init** level to enter initially, using the highest number in the "run_level" field. If the "run_level" field is empty, **init** interprets the run level as **0123456** and enters run level **6**. If there is no *initdefault* entry in *inittab*, then **init** requests an initial run level from the user at boot time.
sysinit	starts the process the first time **init** reads the table and waits for it to terminate before going on to the next *inittab* entry. Entries with the *sysinit* keyword are executed before **init** tries to access the console.

When the run level changes from single-user to a numeric run level (0-6), **init** scans entries with the actions in Table A.3 and executes only those entries with the appropriate "run_level" field set.

Table A-3 inittab run_level field keywords

Keyword	Description
off	sends a warning signal, waits 5 seconds, then sends the kill signal to the process if it is currently running. If the process is not running, it ignores the *inittab* entry.
once	starts the process and continues to the next entry without waiting for the process to complete. When the process dies, **init** does not restart the process.
ondemand	is functionally identical to *respawn*; the *ondemand* keyword is used only with the **a**, **b**, or **c** run-level values.
respawn	starts the process if it is not currently running and continues to the next entry without waiting for it to complete; restarts the process when it dies. If the process is already running, **init** ignores the *inittab* entry.
wait	starts the process and waits for it to complete before going on to the next *inittab* entry.

If the system hardware is capable of detecting power failure and **init** receives a power failure signal, **init** executes entries containing the actions shown in Table A.4 (if the run level is appropriate).

Table A-4 inittab powerfail keywords

Keyword	Description
powerfail	starts the process once and continues to the next entry without waiting for the process to complete.
powerwait	starts the process once and waits for the process to complete before going on to the next entry in the table.

In the following example, the *inittab* entry sets the default run level for the system to single-user mode when **init** is initially invoked:

```
    is:S:initdefault:
```

NOTE The **initdefault** action is not associated with any process. The third colon (:) in this entry is necessary; if it is missing, **init** ignores the entire entry.

You can configure your system to come up in multiuser mode by editing this *inittab* entry, changing the **S** in the run-level field to a **2**.

The following *inittab* entry runs the **/etc/bcheckrc** script when the system is booted (or rebooted) and waits for the process to complete before processing the next entry. (The **bcheckrc** script checks the filesystems and sets the date.)

```
bchk::sysinit:/etc/bcheckrc </dev/console >/dev/console 2>&1
```

The **init** process also starts the **/etc/getty**(M) program according to instructions in *inittab.* The **getty** (get a tty) program sets up communication between the system and terminals. For more information, see the **getty**(M) manual page.

The following example *inittab* entry tells **init** to start a getty process (if one does not already exist) for tty01 (the console) at 9600 baud when the current run level is 2:

```
co:2:respawn:/etc/getty tty01 sc_m
```

The **respawn** action instructs **init** to restart the getty process each time it dies and to continue processing the next *inittab* entry without waiting for the current process to complete. By changing the action from **respawn** to **wait,** you tell **init** to wait until the current process has finished before reading the next entry in *inittab.*

See the **inittab**(F) manual page for a detailed description of the format of the *inittab* file and an explanation of the action keywords.

To make your changes to *inittab* effective immediately, execute the **telinit Q** command. This command causes **init** to reexamine the modified *inittab* file without changing the run level.

Changing scripts in /etc/rc2.d

Upon entering **init** state 2 (multiuser mode) from either a higher **init** state (3-6) or from single-user mode, **init** executes the **/etc/rc2** script according to the instructions in */etc/inittab.* The **rc2** script sets certain environment variables and runs scripts in the */etc/rc2.d* directory. Some of the scripts in *rc2.d* run scripts in subdirectories of the *rc.d* directory. The scripts that **rc2** runs are executed in alphabetic order to ensure proper initialization.

This section describes the scripts in the */etc/rc2.d* directory that are run by **rc2** and explains the steps for adding your own script. The **rc2**(ADM) manual page describes the other scripts that **rc2** runs.

Table A.5 gives a brief description of the scripts in */etc/rc2.d* in the order in which **rc2** executes them.

Table A-5 /etc/rc2.d scripts

Script	Description
S00SYSINIT	starts kernel message logger
S01MOUNTFSYS	mounts filesystems specified in */etc/default/filesys*
S03RECOVERY	tidies up **vi** editing sessions after a crash
S04CLEAN	removes temporary files
S05RMTMPFILES	removes temporary files
S15HWDNLOAD	downloads hardware
S16KERNINIT	starts, process accounting, network, and other kernel initialization
S20sysetup	configures print system and generates */etc/systemid*
S21perf	system accounting
S70uucp	cleans up UUCP lock files
S75cron	starts **cron** daemon
S80lp	starts **lpsched** and net utilities
S86mmdf	starts **mmdf** deliver daemons
S87USRDAEMON	starts user daemons
S88USRDEFINE	executes user-definable commands after boot
S90RESERVED	mails **fsck** output saved during autoboot to *root*

The */etc/rc2.d* directory on your system may contain scripts other than the ones listed in the previous table. The reason for this is that, during installation, many add-on programs insert their own daemon-initialization scripts in this directory. This directory may also include scripts that clean up the temporary or lock files for an add-on program.

You can write your own scripts to run when the system enters **init** state 2. For example, you can write a script that sets up a RAM disk or starts a network and add it to */etc/rc2.d*.

The following factors should be considered when writing a system startup script to be placed in *rc2.d* :

- When going from a higher or lower **init** state to **init** state 2, files that begin with an **S** are executed with the **start** option.
- When changing from a higher **init** state (3-9) to **init** state 2, scripts in *rc2.d* that begin with a **K** are executed with the **stop** option.
- Files that begin with characters other than **S** or **K** are ignored.

- Files are executed in ASCII ascending sort-sequence order; the number in the filename determines the order of execution. In other words, the scripts are executed in the order in which they appear when you execute the **ls** command in */etc/rc2.d*.

  Table A.6 gives, in sort-sequence order (left to right), the ASCII characters that are valid for naming files.

  Table A-6 Valid filename characters

#	%	+	,	-	.	0	1	2	3
4	5	6	7	8	9	:	=	?	@
A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	^	_	a	b
c	d	e	f	g	h	i	j	k	l
m	n	o	p	q	r	s	t	u	v
w	x	y							

- Your script may rely on the existence of services or daemons (such as network services, the **cron** daemon, or the print scheduler) started by other scripts. For example, if your script depends upon certain filesystems being mounted, make sure that you name your script so that it runs after the **S01MOUNTFSYS** script. (When you add a new filesystem with **mkdev fs**, the appropriate files are updated with the information necessary to mount the new filesystem when the system enters multiuser mode.)

To add a function to the initialization procedure, follow these steps:

1. Write a script that performs the desired function.
2. Name the file so that it begins with the uppercase letter S (or K), followed by a two-digit number indicating the order in which it should be executed relative to the other files in the directory, and ends with a name that describes the script's function. For example, **S03RECOVERY** handles crash recovery and is executed after the script that begins with S02 and before the script that begins with S04. You must follow this naming convention to ensure that your script is executed at the proper time.
3. Copy the script into the */etc/rc2.d* directory so that it is executed by **rc2** when the system enters (or leaves) multiuser mode.

If the function that you want to add is in the same category as functions performed by a script already located in */etc/rc2.d*, simply edit the existing script to add the new function. For example, you can add a function related to UUCP to the file *S70uucp*. You can also edit any script to tailor it to your needs. For example, to start process accounting, remove the appropriate comments from the *S16KERNINIT* file. Remember to back up the original script before modifying it.

Starting daemons on a trusted system

If your system is configured in "High" or "Improved" (C2) security mode, all processes must be stamped with an LUID (login user ID) in order to run properly. If you add any scripts that call a **setuid** or **setgid** (set user ID or group ID) program to the */etc/rc* structure, you must remember to set the LUID. This ensures that the system accurately records who does what, even if the effective identity of the user changes. An attempt to run a **setuid** or **setgid** program without an LUID fails with the error message: "cannot execute". If the program does not change the ID of the process, this procedure is unnecessary.

To set the LUID, use the **su**(C) command with this form:

su ***username*** -c "***command***"

where ***username*** is the name of the user or account and ***command*** is the complete command that you want to execute.

For example, the **deliver** daemon checks periodically for undelivered mail. The daemon is in the file */etc/rc2.d/S86mmdf*. (The backslash may be used to break a long command line; it is optional.)

```
su mmdf -c "/usr/mmdf/bin/deliver -b -T3600 \
            -cuucp -L/usr/mmdf/log/uucp.log"
```

This command sets the LUID to *mmdf*, which is the LUID used to administer the mail routing system. The command is run automatically when the system goes into multiuser mode.

Modifying .profile and .login files

The */etc/profile* and */etc/cshrc* files set the default environment for all users of the Bourne and C shells, respectively. The *.profile* and *.login* (for C-shell users) files in the users' home directories contain commands that initialize the environment for each individual user. When a Bourne shell user logs in, the shell first executes the commands in */etc/profile* and then executes the commands in the *.profile* file in the user's home directory. When a C-shell user logs in, the shell executes the commands in */etc/cshrc* and then the commands in the user's *.login* file in their home directory. (Depending on the login shell, other files may apply.)

The files contain commands that set various system variables (for example, **TERM**, **PATH**, and **MAIL**). These variables give the system information such as what terminal type is being used, where to look for programs that the user runs, where to look for the user's mailbox, and what keys to expect for the "kill" and "backspace" functions. For more information about these environment variables, see the chapters about the shells in the *User's Guide*.

There is one *.profile* and/or *.login* file for each user account on the system. The files are placed in the user's home directory when the account is created. Users can modify their own *.profile* or *.login* files or allow the system administrator to make modifications. In either case, these files are ordinary text files and can be modified using a text editor; commands can be added or removed as desired.

Changing the /etc/motd file

The message of the day file, */etc/motd,* contains the greeting displayed whenever a user logs in. Initially, this file contains the name and version number of the operating system. */etc/motd* is an ordinary text file, so the system administrator can change the message by editing the file with a text editor. In general, you should limit the size of the */etc/motd* file to include no more than a single screen of information.

You can modify this file to include messages such as a reminder to clean up directories to preserve disk space, a notice of the next periodic backup, a description of the latest system upgrade, or information about upcoming scheduled system down times. When you announce an unscheduled system shutdown with **wall**, you should edit *motd* to include the shutdown information, so that users who log in after you run **wall** are notified of the impending shutdown.

Appendix B

Authorizing the use of job scheduling commands

This appendix describes how to permit or deny users access to use the job scheduling commands. The **sysadmsh** selection Jobs ➪ Authorize contains the authorization functions. In addition, the **at** and **batch** commands can be regulated by creation of a prototype file that specifies the environment in which the **at** or **batch** commands execute. The commands themselves are described in the chapter entitled "Managing Processes" in the *User's Guide.*

NOTE In addition to granting permission to use these commands, you must make sure that the users have the *chmodsugid* kernel authorization. If you have restricted this authorization, you must grant it as described in the chapter entitled "Administering user accounts".

Changing default authorization for job scheduling

The system is initially configured to permit use of the job scheduling commands by *root, sys, adm* and *uucp* only, denying access to other users.

Changing cron defaults

To change the system default for **cron**, select the following from **sysadmsh**:

Jobs ➪ Authorize ➪ Scheduled ➪ Default

The following three selections are displayed:

None	Execution is not permitted for any users.
Allow	Allow all users to execute the **cron** command.
Deny	Deny all users access to the **cron** command.

The current behavior is highlighted. Use the arrow keys to highlight the behavior desired, or select the first letter. Remember that users can be allowed or denied on an individual basis as well (this is described later). The settings for individuals take precedence over the system defaults.

Changing the at or batch defaults

To change the system default for the **at** and **batch** commands, select the following from **sysadmsh**:

Jobs ⇨ Authorize ⇨ Delayed ⇨ Default

The following three selections are displayed:

None	Execution is not permitted for any users.
Allow	Allow all users to execute the **at** or **batch** command.
Deny	Deny all users access to the **at** or **batch** command.

The currently set behavior is highlighted. Use the arrow keys to highlight the behavior desired, or select the first letter. Remember that users can be allowed or denied on an individual basis as well (this is described later). The settings for individuals take precedence over the system defaults.

Allowing or denying individuals to use cron

To change the default for **cron** for a particular user, select the following from **sysadmsh**:

Jobs ⇨ Authorize ⇨ Scheduled ⇨ User

The cursor is placed on the "User:" field. Enter the name of the user or press ⟨F3⟩ for a list of possible users. When the user name is selected, the following selections are displayed:

Allow	Allow this user to execute the **cron** command.
Deny	Deny this user access to the **cron** command.

Use the arrow keys to highlight the behavior required. This setting overrides the system default.

Viewing user cron permissions

To examine a list of users permitted or denied usage of **cron**, select the following from **sysadmsh**:

Jobs ➪ Authorize ➪ Scheduled ➪ View

If the system default is Allow, a list of users denied access is displayed. If the system default is Deny, a list of users allowed access is displayed.

Allowing or denying individuals to use at or batch

To change the default for **at** or **batch** for a particular user, select the following from **sysadmsh**:

Jobs ➪ Authorize ➪ Delayed ➪ User

The cursor is placed on the "User:" field. Enter the name of the user or press ⟨F3⟩ for a list of possible users. When the user name is selected, the following selections are displayed:

Allow	Allow this user to execute the **at** or **batch** command.
Deny	Deny this user access to the **at** or **batch** command.

Use the arrow keys to highlight the behavior desired. This setting overrides the system default.

Viewing user at or batch permissions

To examine a list of users permitted or denied usage of **at** or **batch**, select the following from **sysadmsh**:

Jobs ➪ Authorize ➪ Delayed ➪ View

As with **cron**, if the system default is Allow, a list of persons denied access is displayed. If the system default is Deny, a list of users allowed access is displayed.

Using environment files for the at or batch commands

It is also possible to define the environment in which **at** and **batch** commands execute. To edit the **at** and **batch** prototype files respectively, use the following **sysadmsh** selections:

Jobs ➪ Authorize ➪ Environment ➪ At

Jobs ➪ Authorize ➪ Environment ➪ Batch

NOTE Only *root* can use these selections.

These options edit the files */usr/lib/cron/.proto* (**at**) or */usr/lib/cron/.proto.b* (**batch**). These files are placed at the start of the shell script formed for all **at** and **batch** jobs. This script must conform to the usual */bin/sh* syntax and contain some variables particular to the prototype file. These variables are:

$d	This is the current directory of the user at the time of submission.
$l	This is the **ulimit** for the user at the time of submission.
$m	This is the **umask** for the user at the time of submission.
$t	This is the time (in seconds past Jan 1, 1970) that the script is run.
$<	This is replaced with the entire script that the user submits. Normally, this appears last in the file, after the prologue that you set up. If you decide to include information after this variable, the shell script may exit before reaching it.

Only the super user can edit these files.

Example of environment file usage

There are many uses for prototype files; two examples are shown below:

- Run jobs in a particular queue at a lower priority by inserting a **nice**(C) command:

```
nice -5 /bin/sh << 'END_OF_FILE'
$<
END_OF_FILE
```

- Specify that a queue should execute commands using an alternative shell:

```
/bin/csh << 'END_OF_FILE'
$<
END_OF_FILE
```

For most sites, the prototype files provided with the distribution should be sufficient.

Appendix C

Using the system console and color displays

This appendix is concerned with utilities and features that affect the use of the system console and other color displays. Console displays are connected to a standard display adapter, while color terminals (such as Sun River terminals) are connected to the system by special adapters.

This chapter explains how to do the following:

- Set or change the console keyboard type (XT or AT) using **kbmode**(ADM)
- Protect the console from excessive wear by blanking the screen when not in use
- Use the **multiscreen**(M) facility to control multiple screens from a single display
- Change the font used on the screen display using **vidi**(C)
- Set the colors displayed on color screens using **setcolor**(C)

If you wish to set up a serial console, refer to "Setting up a serial console" in the chapter entitled "Administering serial terminals."

Console keyboard type selection

The operating system supports two keyboard modes: AT and XT. By default, the system is configured for use with an XT keyboard. This is because an XT (or other non-AT) keyboard will not work in AT mode; the system will not recognize keyboard input. An AT keyboard will work properly in XT mode, but the extended keyset found on the AT 101 or 102 key keyboard is not accessible. Therefore if you have an AT keyboard you should reset the keyboard mode to AT, in order to make full use of the extended keyset. The **kbmode**(ADM) utility is used to test and set the keyboard mode.

Some keyboards have an AT keyboard layout, but do not support AT mode. To test your keyboard to determine if it supports AT mode, invoke **kbmode** with the test option as follows:

kbmode test

A sample session with **kbmode** in test mode is shown below, complete with user input in boldface:

```
# kbmode test
Current keyboard mode is XT
Do you want to determine if your keyboard supports AT mode? y
During the test the keyboard will be put into AT mode.
You should then press the space bar two or three times.
Are you ready to start? y
Please hit the space bar now!

The keyboard has been returned to its default mode.
It supports AT mode.
#
```

The display will be temporary initialized to AT mode.

Switching keyboard modes manually

The **kbmode** utility is also used to set the mode. Use one of the following commands for switching to AT and PC/XT mode, respectively:

kbmode at

kbmode xt

Changing modes permanently

To change the default mode permanently, the kernel parameter **KBTYPE** must be set to the proper keyboard. To change **KBTYPE**, run the **configure**(ADM) utility and select option 13, "Hardware Dependent Parameters." Change the parameter value and relink the kernel as described in "Reallocating kernel resources with configure" in the chapter entitled "Tuning system performance."

Using the console screen protection feature

The console can be set up to blank after a certain number of seconds to protect the screen from excessive wear. (This is similar to a feature available with most terminals.) The kernel parameter **TBLNK** controls the console screen protection feature. By default, screen blanking is not performed: to enable this feature, you must invoke the **sysadmsh** selection

System ⇨ Configure ⇨ Kernel ⇨ Parameters.

Select category 6: "MultiScreens", and change the value of **TBLNK** to the number of seconds that the system should wait before blanking the screen. The kernel must then be relinked and booted for the new behavior to take effect. Use the **sysadmsh** System ➪ Configure ➪ Kernel ➪ Rebuild selection to relink the kernel. For complete instructions, see the section "Reallocating kernel resources with configure" in the chapter entitled "Tuning system performance."

Using MultiScreen

With Multiscreen, you can use your console as several terminals at one time. Pressing a simple key combination switches you from one screen to another, with each screen acting as an independent terminal.

Each multiscreen is independent, which means that you can log in and run programs on each screen. Output from your programs is saved in a screen buffer, so you see the most recent output for whichever screen you look at. If you stop output to one screen, as when you press the ⟨Ctrl⟩s key combination, only that screen is affected.

The amount of memory in your computer determines the number of multiscreens available on your system. When you boot your system, the number of automatically-enabled multiscreens is displayed. Most machines have between two and six multiscreens enabled, but your machine can have up to twelve if your system has sufficient memory. To increase the number of multiscreens on your system, you need to add to your system's memory; additional screens will be enabled automatically.

Although all of the multiscreens can be open and active at once, you see only one screen at a time. The selected multiscreen is like a terminal that is "connected" to the keyboard. Switching between screens is like moving to another terminal because each multiscreen has its own device file.

The multiple screen feature uses the */dev/tty[01...12]* device files. These files provide character I/O between your system and your computer screen and keyboard.

To select any active screen, press ⟨Alt⟩-F*n*, where **F*n*** is one of the function keys on your keyboard. Function keys are generally located across the top or down the far left side of the keyboard. The tty01 is the ⟨Alt⟩⟨F1⟩ terminal, tty02 is the ⟨Alt⟩⟨F2⟩ terminal, tty03 is ⟨Alt⟩⟨F3⟩, etc. For example, the following keystroke switches you to screen 6, corresponding to */dev/tty06*:

```
⟨Alt⟩⟨F6⟩
```

You can also rotate through the screens by pressing the Control and Print Screen key combination, ⟨Ctrl⟩⟨PrtSc⟩ (using the ⟨Ctrl⟩ key and the ⟨PrtSc⟩ key). Use this combination to access screens for which you do not have function keys. For example, if you have twelve multiscreens enabled, but your computer keyboard has only ten function keys, display screen eleven by pressing ⟨Alt⟩⟨F10⟩ to get to screen 10, and then pressing ⟨Ctrl⟩⟨PrtScr⟩ to rotate to screen 11. To access screen 12, press ⟨Ctrl⟩⟨PrtScr⟩ again. Pressing ⟨Ctrl⟩⟨PrtScr⟩ again rotates you back to the first multiscreen, *tty01*.

Note that you can use ⟨Ctrl⟩⟨Alt⟩ ***function-key*** combinations in addition to ⟨Alt⟩ ***function-key*** combinations to change multiscreens. This is especially useful in applications that reserve the ⟨Alt⟩ ***function-key*** combinations for their own use. This can be configured using the **mapkey**(ADM) utility.

For more information, refer to **multiscreen**(M) and **screen**(HW).

Reducing the number of multiscreens

The system is configured with 12 **multiscreen**(M) console screens by default. Although this does not significantly affect performance, you can reduce the number of screens if desired. To configure the system for fewer screens, do the following:

1. Log in as *root* and enter the **sysadmsh**(ADM).
2. Use the System ➪ Configure ➪ Kernel ➪ Parameters selection and select category 6, Multiscreens.
3. Skip the parameters displayed by pressing ⟨Return⟩ until you reach the **NSCRN** parameter. Enter a value corresponding to the number of screens you wish to enable.
4. Calculate the amount of screen memory in Kbytes (controlled by the **SCRNMEM** parameter) as follows:

   For 25-line displays:

   SCRNMEM = 10K + 4K * **NSCRN**

   For 43-line displays:

   SCRNMEM = 10K + 8K * **NSCRN**

5. Enter a value for **SCRNMEM** to match the value you calculated in the previous step.
6. Exit the **configure**(ADM) menu by entering **q** and pressing ⟨Return⟩.
7. Use the System ➪ Configure ➪ Kernel ➪ Rebuild selection to build the new kernel with the revised screen values. (The Rebuild should be visible when you exit the previous selection.) Follow the prompts. Exit **sysadmsh** when the process is complete.

8. You must then disable the unused **multiscreen** ttys. For example, if you reconfigured for 5 screens after having 12, you would enter the following command at the system prompt to disable ttys 6 through 12:

   ```
   disable tty06 tty07 tty08 tty09 tty10 tty11 tty12
   ```

9. You should now shut down the system and reboot from the new kernel. Enter the following command:

   ```
   shutdown -g0
   ```

   When the reboot message is displayed, press ⟨Return⟩ to restart the system.

Multiscreens and multiple video adapters

Video adapters can be assigned to **multiscreens** dynamically. All start on the primary adapter, but any screen can be moved to another video card with the **vidi**(C) command.

Valid adapter names are "mono", "cga", "ega", and "vga".

For example, if your primary video adapter is an EGA and you have a MONO secondary adapter, you can move the current screen onto the MONO card using the following command:

```
vidi mono
```

Changing video fonts

You can display the full range of characters on a display adapter by using the **vidi**(C) utility. Normally, if you have a console with a display adapter which has a character set defined in a ROM, you will be able to display only those characters defined in that ROM. In addition, in order to display the entire font set, the **mapchan** file for the console must correspond to the character set defined in that ROM.

In addition to using **vidi**(C) to override ROM, you can use it to define certain display fonts on some display adapters. For example, the VGA adapter will allow you to display fonts in the sizes 8x8, 8x14, and 8x16.

The **vidi**(C) utility defines the font for one of these six character sets. Table C.1 lists the font definition files in the directory */usr/lib/vidi.*

Table C-1 Font definition files

Character Set	8x8 font	8x14 font	8x16 font
PC standard	font8x8	font8x14	font8x16
ISO 8859/1	iso.8x8	iso.8x14	iso.8x16
PC Nordic	nor.8x8	nor.8x14	nor.8x16
PC Portuguese	por.8x8	por.8x14	por.8x16
PC Spanish	spa.8x8	spa.8x14	spa.8x16
PC Greek	grk.8x8	grk.8x14	grk.8x16

Controlling color displays with setcolor

setcolor(C) is a simple utility that enables you to control the colors used on the display screen. (The **setcolor** command usually has no effect on monochrome displays or terminals.) Both foreground and background colors can be set independently in a range of 16 colors. **setcolor** can also set the reverse video and graphics character colors.

The following colors are available:

blue	magenta	brown	black
lt_blue	lt_magenta	yellow	gray
cyan	white	green	red
lt_cyan	hi_white	lt_green	lt_red

To display these colors, simply invoke **setcolor** without options.

The following flags are available. In the arguments below, "color" is taken from the above list.

Changing the foreground and background colors

You can set both background and foreground colors with a single command as in the following example:

```
setcolor red white
```

This results in red characters on a white background. If only one color is specified, the foreground color is changed. To change only the background color, use the **-b** option, as in the following:

```
setcolor -b red
```

This changes the background color to red.

Changing reverse video colors

Reverse video normally inverts the foreground and background colors. **setcolor** allows you to set these independently. For example:

```
setcolor -r blue red
```

This command sets the foreground reverse video color to blue and the background reverse video color to red.

Changing the screen border color

For CGA cards you can also change the color of the square border that defines the text region of the display:

```
setcolor -o green
```

The above example changes the border to green without affecting the rest of the display.

Sounding the keyboard bell

One of the less obvious functions of **setcolor** is to control the sound of the bell that is usually built into the display or keyboard. To change the bell tone, you must supply a pitch and duration. (Pitch is the period in microseconds, and duration is measured in fifths of a second.) When using this option, a ⟨Ctrl⟩**g** (bell) must be echoed to the screen for the command to work. For example:

```
setcolor -p 500 2
echo ^G
```

This command sets the bell to a high pitch of short duration. The higher the pitch number, the lower the sound generated. For example, this command sets the bell to a sustained low tone:

```
setcolor -p 7000 8
```

Note that each time ⟨Ctrl⟩**g** is pressed, the bell will sound the tone most recently set.

Resetting the screen

The **-n** option returns the screen to "normal" white characters on black background.

Appendix D

UNIX directories and special device files

This appendix lists the most frequently used files and directories on a UNIX system. Many of these files and directories are required for proper system operation and must not be removed or modified. The following sections briefly describe each directory.

This appendix also describes device nodes relating to filesystems and terminals. For a full description of the special files mentioned here, see the manual pages in the Hardware Dependent (HW) section.

UNIX directories

The following subsections discuss each of the main directories of the operating system.

The root directory

The root directory (/) contains the following system directories:

/bin	UNIX command directory
/dev	device special directory
/etc	additional program and data file directory
/lib	C programming library directory

/mnt	mount directory (reserved for mounted filesystems)
/usr	user service routines (may contain user home directories)
/tcb	system files that are part of the TCB (Trusted Computing Base)
/tmp	temporary directory (reserved for temporary files created by programs)

All of the above directories are required for system operation.

The root directory also contains a few ordinary files. Of these files, the most notable is the */unix* file, which contains the UNIX kernel image.

The /bin directory

The */bin* directory contains the most common UNIX commands, that is, the commands likely to be used by anyone on the system.

The /dev directory

The */dev* directory contains special device files that control access to peripheral devices. All files in this directory are required, and must not be removed.

There are several subdirectories to the */dev* directory. Each of these subdirectories holds special device files related to a certain type of device. For example, the */dev/dsk* directory contains device files for floppy and hard disks. The operating system supports both XENIX and UNIX device naming conventions. Where appropriate, the files in the */dev/dsk* directories are linked to the device files that exist in */dev*. You can access the same device through the file in */dev* or the file for the same device in a subdirectory of */dev*.

Table D.1 contains a list of the more common devices.

Table D-1 /dev device nodes

UNIX Device	XENIX Device	Name
/dev/console	same	system console
/dev/rdsk/*	/dev/r*	raw devices
/dev/dsk/0s0	/dev/hd00	entire disk on drive 0
/dev/dsk/0s1	/dev/hd01	first disk partition on drive 0
/dev/dsk/0s2	/dev/hd02	second disk partition on drive 0
/dev/dsk/1s0	/dev/hd10	entire disk on drive 1
/dev/dsk/1s1	/dev/hd11	first disk partition on drive 1
/dev/dsk/1s2	/dev/hd12	second disk partition on drive 1
/dev/dsk/f05d9	/dev/fd048ds9	360K floppy drive 0
/dev/dsk/f05q	/dev/fd096ds9	720K floppy drive 0
/dev/dsk/f05h	/dev/fd096ds15	1.2MB floppy drive 0
/dev/dsk/f03h	/dev/fd0135ds18	1.44MB floppy drive 0
/dev/lp	same	lineprinter
/dev/kmem	same	kernel virtual memory
/dev/mem	same	physical memory
/dev/null	same	null device
/dev/rmt0	/dev/rct0	default cartridge tape device
-	/dev/rft0	QIC-40 tape device
-	/dev/rctmini	minicartridge tape device
/dev/root	same	root file structure
/dev/swap	same	swap area
/dev/ptypnn	-	pseudo tty (master)
/dev/ttypnn	-	pseudo tty (slave)
/dev/ttynn	-	console multiscreen

The /etc directory

The */etc* directory contains miscellaneous system program and data files. All files are required, but many can be modified. The data files in the directories */etc/rc.d* and */etc/rc2.d* contain initialization commands run by the **/etc/rc2** script when the system goes into multiuser mode. (The */etc/rc* directories are discussed extensively in Appendix A, "Customizing system startup.")

The data files in the directory */etc/default* contain default information that is used by system commands (see **default**(F)). The data files shown in Table D.2 may be modified, but may not be removed.

Table D-2 /etc/default files

File	Utility
/etc/default/archive	**sysadmsh**(ADM) backup default information
/etc/default/authsh	**sysadmsh**(ADM) default account information
/etc/default/backup	**backup**(ADM) default information
/etc/default/boot	**boot**(ADM) information
/etc/default/cleantmp	**cleantmp**(ADM) default information
/etc/default/cron	**cron**(C) default logging information
/etc/default/device.tab	**pkgadd**(ADM) default information
/etc/default/dumpdir	**xdumpdir**(ADM) default information
/etc/default/filesys	**sysadmsh**(ADM) default filesystem data
/etc/default/format	**format**(C) default information
/etc/default/goodpw	**goodpw**(ADM) default password check data
/etc/default/idleout	**idleout**(ADM) default information
/etc/default/lang	default locale information
/etc/default/lock	**lock**(C) default information
/etc/default/login	**login**(M) default information
/etc/default/lpd	**lp**(C) default information
/etc/default/man	**man**(C) online man page default information
/etc/default/mapchan	**mapchan**(M) default information
/etc/default/mapkey	**mapkey**(M) default information
/etc/default/micnet	**micnet**(C) default information
/etc/default/msdos	location of DOS disks (A:, B:,. . .) for **dos**(C) commands
/etc/default/passwd	**passwd**(C) default information
/etc/default/purge	**purge**(C) default information
/etc/default/restor	**xrestore**(ADM) default information
/etc/default/scsihas	defines SCSI host adaptors
/etc/default/slot	defines EISA slots (EISA machines only)
/etc/default/su	**su**(C) default information
/etc/default/tape	**tape**(C) default device information
/etc/default/tar	**tar**(C) default device information
/etc/default/usemouse	**usemouse**(C) default information

The /lib directory

The */lib* directory contains runtime library files for C and other language programs. This directory is required.

The /mnt directory

The */mnt* directory is an empty directory reserved for mounting removable filesystems.

The /usr directory

The /usr directory consists of several subdirectories that contain additional UNIX commands and data files. It is also the default location of user home directories.

The */usr/bin* directory contains more UNIX commands. These commands are used less frequently or are considered nonessential to UNIX system operation.

The */usr/include* directory contains header files for compiling C programs.

The */usr/lib* directory contains more libraries and data files used by various UNIX commands.

The */usr/spool* directory contains various directories for storing files to be printed, mailed, or passed through networks.

The */usr/tmp* directory contains more temporary files.

The */usr/adm* directory contains data files associated with system administration and accounting. In particular, the */usr/adm/messages* file contains a record of all error messages sent to the system console. This file is especially useful for locating hardware problems. For example, an unusual number of disk errors on a drive indicates a defective or misaligned drive. Because messages in the file can accumulate rapidly, the file must be deleted periodically.

The /tcb directory

The */tcb* directory contains all files that are part of the TCB (Trusted Computing Base). These files comprise the security enhancements made to the operating system to make it more secure than other UNIX operating systems. The security features are discussed in the chapter entitled "Maintaining system security."

The /tmp directory

The */tmp* directory contains temporary files created by UNIX programs. The files are normally present when the corresponding program is running, but may also be left in the directory if the program is prematurely stopped. You can remove any temporary file that does not belong to a running program.

Log files

A variety of directories contain log files that grow in size during the normal course of system operation. Many of these files must be periodically cleared to prevent them from taking up valuable disk space. (See the section on "Checking and clearing log files" in the chapter entitled "Managing filesystems.") Table D.3 lists the log files (by full pathname) and their contents.

Table D-3 System log files

Filename	Description
/etc/ddate	records date of each backup.
/usr/adm/pacct	records accounting information; grows rapidly when process accounting is on. (See **accton**(ADM) and **acctcom**(ADM).)
/usr/adm/messages	records error messages generated by the system when started. (See **messages**(M).)
/etc/wtmp	records user logins and logouts. (See **login**(M).)
/usr/adm/sulog	records each use of the **su** command; grows only if option is set in the */etc/default/su* file. You must create */etc/default/su*. (See **su**(C).)
/usr/lib/cron/cronlog	records each use of the **at**(C) and **cron**(C) commands.
/usr/spool/uucp/.Log/utility/sitename/*	logs UUCP commands used over a UUCP network. The ***utility*** and ***sitename*** are the name of the UUCP utility and the name of the remote site, respectively.
/usr/spool/uucp/.Log/.Old/*	stores old log files placed in this directory by the **uudemon.clean** shell script.

Special device files

Many of the filesystem maintenance tasks described in this guide require the use of special filenames, block sizes, and gap and block numbers. The following sections describe each in detail.

Special filenames

A special filename is the name of either the device special block or character I/O file, which corresponds to a peripheral device such as a hard or floppy disk drive. These names are required in such commands as **mkfs**(ADM), **mount**(ADM), and **df**(C) to specify the device containing the filesystem to be created, mounted, or searched.

Table D.4 lists the XENIX and UNIX special filenames and corresponding devices for hard and floppy disk drives on a typical computer.

Table D-4 Disk device filenames

Filename	Disk Drive
/dev/fd0	floppy drive 0
/dev/dsk/f0	floppy drive 0
/dev/fd1	floppy drive 1
/dev/dsk/f1	floppy drive 1
/dev/hd00	entire hard disk
/dev/dsk/0s0	entire hard disk
/dev/root	root filesystem
/dev/u	user filesystem

Block sizes

The block size of a disk is the number of blocks of storage space available on the disk, where a block is 1024 bytes of storage. Most commands, however, report disk space in terms of 512 byte blocks, in particular **df**(C), **du**(C), **ls**(C), **lc**(C), and **find**(C). A 500-byte file on a 1024-byte block filesystem is reported as using 2 blocks by these utilities, as the file uses one physical disk block that is equivalent to two 512-byte filesystem blocks. The size of a 40-megabyte hard disk in 1024-byte blocks is 39168. Note that some of the blocks on the disk are reserved for system use and cannot be accessed by user programs. The block size of a typical floppy disk depends on the total storage capacity of the disk, as given by the manufacturer.

Gap and block numbers

The gap and block numbers are used by the **mkfs**(ADM) and **fsck**(ADM), commands to describe how the blocks are to be arranged on a disk. Table D.5 lists the gap and block numbers for the floppy and hard disks used with a typical computer.

Table D-5 Gap and block numbers

Disks	Gap	Block
floppy disk, 48ds9	1	9
floppy disk, 96ds15	1	15
floppy disk, 135ds9	1	9
floppy disk, 135ds18	1	18
hard disk	1	34

The number of blocks can also be determined by multiplying the number of sectors per track (for example, 17) by the number of heads on the hard disk, dividing by 2 (because there are 2 sectors per block), and rounding down to the nearest integer.

Terminal and network requirements

The **enable**(C) and **disable**(C) commands enable and disable logins on terminals. **enable** and **disable** require the names of the serial lines through which a terminal or network is to be connected. Table D.6 lists the device special filenames of the two serial lines (actually two serial ports either with or without modem control). The character I/O files corresponding to these serial lines can be found in the */dev* directory. Note that the files */dev/console* and */dev/tty01* through */dev/tty12* represent "hardwired" devices and are not available for connection to terminals or hardware. Also, refer to **serial**(HW) for more information on serial lines.

Table D-6 Serial devices

Filename	Line
/dev/tty1a	main serial line (without modem control)
/dev/tty2a	alternate serial line (without modem control)
/dev/tty1A	main serial line (with modem control)
/dev/tty2A	alternate serial line (with modem control)

Index

SCO®
OPEN SYSTEMS SOFTWARE

Please help us to write computer manuals that meet your needs by completing this form. Please post the completed form to the Technical Publications Research Coordinator nearest you: The Santa Cruz Operation, Ltd., Croxley Centre, Hatters Lane, Watford WD1 8YN, United Kingdom; The Santa Cruz Operation, Inc., 400 Encinal Street, P.O. Box 1900, Santa Cruz, California 95061, USA or SCO Canada, Inc., 130 Bloor Street West, 10th Floor, Toronto, Ontario, Canada M5S 1N5.

Volume title: ______________________________

(Copy this from the title page of the manual, for example, SCO UNIX Operating System User's Guide)

Product: ______________________________

(for example, SCO UNIX System V Release 3.2 Operating System Version 4.0)

How long have you used this product?

❑ Less than one month ❑ Less than six months ❑ Less than one year

❑ 1 to 2 years ❑ More than 2 years

How much have you read of this manual?

❑ Entire manual ❑ Specific chapters ❑ Used only for reference

	Agree				*Disagree*
The software was fully and accurately described	❑	❑	❑	❑	❑
The manual was well organized	❑	❑	❑	❑	❑
The writing was at an appropriate technical level (neither too complicated nor too simple)	❑	❑	❑	❑	❑
It was easy to find the information I was looking for	❑	❑	❑	❑	❑
Examples were clear and easy to follow	❑	❑	❑	❑	❑
Illustrations added to my understanding of the software	❑	❑	❑	❑	❑
I liked the page design of the manual	❑	❑	❑	❑	❑

If you have specific comments or if you have found specific inaccuracies, please report these on the back of this form or on a separate sheet of paper. In the case of inaccuracies, please list the relevant page number.

May we contact you further about how to improve SCO UNIX documentation? If so, please supply the following details:

Name ____________________ *Position* ____________________

Company ______________________________

Address ______________________________

City & Post/Zip Code ______________________________

Country ______________________________

Telephone ____________________ *Facsimile* ____________________

31 January 1992

BH01207P000

58076

CLUSTER

BJ01205P000

MANUAL

AU01211P000