AT&T

UNIX® SYSTEM V RELEASE 4

System Administrator's Guide

UNIX Software Operation

AT&T

UNIX® SYSTEM V RELEASE 4

System Administrator's Guide

UNIX Software Operation

Published by Prentice-Hall, Inc.
A Division of Simon & Schuster
Englewood Cliffs, New Jersey 07632

IMPORTANT NOTE TO USERS

While every effort has been made to ensure the accuracy of all information in this document, AT&T assumes no liability to any party for any loss or damage caused by errors or omissions or by statements of any kind in this document, its updates, supplements, or special editions, whether such errors are omissions or statements resulting from negligence, accident, or any other cause. AT&T further assumes no liability arising out of the application or use of any product or system described herein; nor any liability for incidental or consequential damages arising from the use of this document. AT&T disclaims all warranties regarding the information contained herein, whether expressed, implied or statutory, *including implied warranties of merchantability or fitness for a particular purpose.* AT&T makes no representation that the interconnection of products in the manner described herein will not infringe on existing or future patent rights, nor do the descriptions contained herein imply the granting or license to make, use or sell equipment constructed in accordance with this description.

AT&T reserves the right to make changes without further notice to any products herein to improve reliability, function, or design.

TRADEMARKS

Epson FX-86e is a trademark of Epson America, Inc.
HP and LaserJet are registered trademarks of Hewlett-Packard Company.
PostScript is a registered trademark of Adobe Systems, Inc.
Proprinter is a trademark of International Business Machines.
UNIX is a registered trademark of AT&T.
XENIX is a registered trademark of Microsoft Corporation.

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-947086-7

UNIX PRESS
A Prentice Hall Title

P R E N T I C E   H A L L

ORDERING INFORMATION

UNIX® SYSTEM V, RELEASE 4 DOCUMENTATION

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Simon & Schuster Asia Pte. Ltd., *Singapore*
Editora Prentice-Hall do Brasil, Ltda., *Rio de Janeiro*

Contents

Figures and Tables

About This Document

Introduction

This book has been designed to help you do the work of a system administrator for a computer running UNIX System V Release 4.0 on an AT&T 3B2 Computer. You may be the owner of a small business, personally maintaining and overseeing the operations of a single 3B2 Computer. Or you may be an administrator for a large organization in which many users share a network of 3B2 Computers. In either case, this guide will help you install and maintain various services on your system, and serve the needs of your users.

Among the new features introduced in UNIX System V Release 4.0 are many new software tools for administration, including a new version of the system administration menus. These tools will help you install your machine and software, set up the resources and environments that best fit the needs of your users, do routine maintenance procedures, and provide emergency troubleshooting service.

How This Guide Is Organized

This guide has been designed to allow you to find all the information you need about a particular area of administration in one place. Each chapter covers a discrete administrative function, such as file system administration, security, or the backup service. In addition, appendixes, a glossary, and an index are provided to make this guide easy to use and understand.

Organization of the Chapters

The chapters are arranged in alphabetical order, like the topics of administration presented on the main system administration menu (the menu that appears on your screen when you enter the sysadm command).

Organization of Each Chapter

Each chapter describes the software associated with a function, and provides instructions for performing that function. For some functions, UNIX System V Release 4.0 provides a user-friendly menu interface that can help you do administrative functions without using UNIX system shell commands. This interface is accessed through the sysadm command. For functions for which this interface is available, you will see instructions for invoking the appropriate menu at the beginning of the relevant chapter. Because the menus (and other screen messages provided with them) are self-explanatory, detailed instructions about how to use a menu that you have accessed are not included in each chapter.

Instead, Appendix C of this guide, "Using the sysadm Interface," provides a sample walk-through for one menu, and defines all the components of the menu system. In addition, the interface itself provides on-line "help messages" that you can access while using the menus through the sysadm command.

Each chapter provides instructions for accessing the appropriate sysadm menu, and a table listing those shell commands that can be used in place of menu options.

Notation Conventions Used in This Guide

This section describes the notation conventions used in this book.

- References to literal computer input and output (such as commands entered by the user or screen messages produced by the system) are shown in a `monospace` font, as in the following example:

```
$ ls -l report.oct17
-rw-r--r--   1 jim    doc      3239 May 26 11:21 report.oct17
```

- Substitutable text elements (that is, text elements that you are expected to replace with specific values) are shown in an italic font, as in the following example:

  `$ cat` *filename*

  The italic font is a signal that you are expected to replace the word *filename* with the name of a file.

- Comments in a screen display–that is, asides from the author to the reader, as opposed to text that is not computer output–are shown in an *italic* font and are indented, as in the following example:

```
                 .
                 .
                 .
      command interaction
                 .
                 .
                 .
Press RETURN to continue.
```

- Instructions to the reader to type input usually do not include explicit instructions to press the RETURN key at the appropriate times (such as after entering a command or a menu choice) because this instruction is implied for all UNIX system commands and menus.

In one circumstance, however, an instruction to press the RETURN key is explicitly provided: when, during an interactive routine, you are expected to press RETURN without having typed any text, an instruction to do so will be provided, as follows:

```
Type any key to continue: RETURN
$
```

- Control characters are shown by the string CTRL-*char* where *char* is a character such as "d" in the control character CTRL-d. To enter a control character, hold down the CTRL key and press the letter shown. Be sure to type the letter exactly as specified: when a lower case letter is shown (such as the "d" in the example above), enter that lower case letter. If a character is shown in upper case (such as CTRL-D), you should enter an upper case letter.
- The system prompt signs shown in examples of interactive sessions are the standard default prompt signs for AT&T UNIX System V Release 4.0:
  - the dollar sign (`$`) for an ordinary user
  - the pound sign (`#`) for the owner of the `root` login

How to Use This Guide

This book has been designed to help anyone doing administrative tasks on a computer running UNIX System V Release 4.0. Specifically, it has been written to help you understand the job of an administrator and find out exactly how to set up, configure, and maintain UNIX System V Release 4.0 on a 3B2 Computer.

The guide assumes that you know how to enter commands at a computer terminal, and that you have an awareness of such UNIX system fundamentals as the directory structure and the shell. You should also feel comfortable using the computer itself; you should know how to turn it on and how to install peripherals (such as modems, terminals, and printers) for it. To familiarize yourself with your computer and set it up for administration, see your computer installation manual and the documentation about the peripherals that came with your computer. You may also want to refer to the *Product Overview and Master Index* for descriptions of other documents about the UNIX system that might be helpful to you as an administrator.

New Administrators

If you have no experience as a UNIX system administrator, use this guide as a textbook for the work you are undertaking. Begin by reading Chapter 1, "Overview of System Administration." This chapter describes the duties of an administrator, suggests how to organize those duties, and tells you where, in this guide, to find more information about each of those duties.

Then you can read individual chapters to learn about those areas of administration with which you need to familiarize yourself. All administrators need to do many of the tasks described in this book. Some activities, however, may or may not be required for your system, depending on your site, your resources, and your customers. Read those sections of this book that are useful for your needs.

Experienced Administrators

If you are an experienced administrator and you know what information you need to gather and what questions you need to have answered, use this guide as a reference book. You will probably be faced with a task for which you want detailed instructions. Begin your search for the instructions by perusing the table of contents (and, if necessary, the index) for the topic you need. To find out whether there's a menu interface available for your task, see the first page of the appropriate chapter or look in Appendix C, "Using the `sysadm` Interface." This appendix contains a complete list of the menus and tasks included in the system administration menu interface.

If You Use the Menus

If you decide to use the menu interface when you do a particular task, you'll need to find out how to access it. See the first page of the chapter that discusses the area of administration associated with your task. Once you've accessed the appropriate menu and you want to find out how to use it, see Appendix C, "Using the `sysadm` Interface."

If You Do Not Use the Menus

If you decide not to use the menu interface, continue reading the chapter until you find instructions for the administrative task you want to perform. These instructions will specify the running of shell commands.

1 Overview of System Administration

Introduction

NOTE Most of the work described in this book can be done only by a user who is logged in as `root`. Therefore you must use this login name whenever you're going to do administrative tasks. The explicit instruction to log in as `root` is not included in every administrative procedure; we assume, throughout this book, that you have logged in as `root`.

This chapter presents a general "job description" for you, the system administrator: it describes, in broad terms, the tasks for which you are responsible, and lists the documentation in which you can find procedures for doing those tasks. Almost all the procedures in this book can be done by issuing shell level commands. The descriptions and procedures presented in each chapter specify the appropriate commands.

For many types of work, however, you have a choice of working on the shell command line or working with an interface composed of menus and forms for system administration. Because these menus and forms are invoked by executing the sysadm command, we refer to them collectively as "the sysadm interface."

On the following page you'll find a one-page summary of the commands you need to know to start using the sysadm interface. For a more detailed description of how to use these menus and forms, see Appendix C of this guide, "Using the sysadm Interface."

Quick Reference to the sysadm Interface

Function Keys

The main tool for manipulating the interface is a set of eight function keys. Labels highlighted in reverse video at the bottom of the screen show the function assigned to each; the functions assigned to some keys change for different types of frames, but F1 is always mapped to HELP.

If your function keys do not seem to work, you can simulate them using the two-character sequences CTRL-f 1 through CTRL-f 8. The CANCEL function key dismisses the current frame (except for the main menu, which cannot be canceled). The CMD-MENU function key provides a Command Menu of other useful commands.

Menus

To move between menu items, use the down arrow ↓ and up arrow ↑ keys. To select a menu item, use the ENTER key or the ENTER function key.

Forms

To move to the next field, use the TAB key or arrow keys. After filling in a form, press the SAVE function key to process the data entered.

Text Frames

A text frame contains more than one logical page of text if the scroll bar on the right frame border contains a caret ^ at the top or a v at the bottom; use the NEXTPAGE and PREVPAGE function keys to move between these pages.

Command Line

To go to the command line, use the CTRL-j or CTRL-f c character sequence. Any command from the Command Menu can be typed directly here; press ENTER to process the command and return to the current frame. Use the `refresh` command to redraw a corrupted screen and the `cleanup` command to dismiss most frames from a cluttered screen.

Exiting from sysadm

To exit from the sysadm interface, press the COMMANDS function key and select the `exit` item, or go to the command line and type `exit` ENTER. (The CANCEL function key is not equivalent to `exit`.)

See Appendix C for complete information on using the UNIX System V Release 4.0 sysadm interface.

The Job of the Administrator

The job of a system administrator is to provide and support computer services for a group of users. Specifically, it's the administrator's job to do the following:

- set up the computer system, including hardware and software
- allocate resources among users
- optimize the use of software resources
- protect software resources
- do routine maintenance chores
- repair defective hardware and software as problems arise

The rest of this section describes the specific tasks associated with each of these broadly defined areas of responsibility.

Setup of Hardware and Software Resources

The checklist below summarizes the steps you need to take when setting up your computer for the first time. When a reference contains a chapter title without a book title, the reference is to a chapter in this book.

Step	Task	Documentation
1	Install the computer	Your computer installation manual
2	Install, connect, and set up the console terminal	Your computer installation manual, your terminal manual, and the "System Setup" chapter
3	Install and connect the console printer	Your terminal manual and your printer manual
4	Install the Essential Utilities	*Source Code Product Build Instructions*
5	Complete the initial setup procedure	"System Setup" chapter
6	Install software packages (optional)	"Software Management" chapter

7	Set up ports	"Service Access" chapter
8	Install peripheral devices (optional) Connect printers and install the LP print service (optional)	"Storage Device Management" chapter 'Print Service" chapter and your printer installation manual
9	Customize the system profile (optional)	"User and Group Management" chapter
10	Create groups for users (optional)	"User and Group Management" chapter
11	Assign user logins and passwords	"User and Group Management" chapter
12	Set up a network (optional)	*Network User's and Administrator's Guide* and the "Network Services" chapter

The rest of this section describes the tasks shown in the table and lists the books in which you can find instructions for them.

Steps 1-3: Install the Computer, Console Terminal, and Console Printer

Your first task is the physical installation of your computer, the console terminal, and, if you're planning to have a dedicated printer for the console terminal, the console printer. Begin by installing your computer, following the instructions in the installation manual delivered with it.

Next, install the console terminal and connect it to your computer, as instructed in the terminal manual. Turn on the terminal and set the options for it, as described in the "System Setup" chapter of this book.

To make record keeping more convenient, you may want to hook up a printer to the console terminal for use exclusively by you. If you decide to do this, set up your console printer now, following the instructions in your printer installation manual.

Power up the computer according to the instructions in the "System Setup" chapter.

To protect your system from unauthorized use, we recommend that the first thing you do once your computer is running is make a "floppy key" and change the firmware password for your computer. (The machine is delivered with a default password.) Instructions for these tasks are in the "System Setup" chapter.

Step 4: Install the Essential Utilities

Because the connections between your computer and peripheral devices must be made through the software as well as through the hardware, it's a good idea to install the Essential Utilities (the basic UNIX system software) next. These utilities are delivered on a set of floppy diskettes or cartridge tapes. For installation instructions, see the *Source Code Product Build Instructions*.

Step 5: Complete the Initial Setup Procedure

Once you have physically installed the hardware and are running the Essential Utilities on your computer, you need to complete a procedure that will involve answering several questions, such as the following:

- What is the name of this computer?
- If this computer is going to be part of a network, what is its node name?
- What's today's date? What's the current time?

The information provided in your answers will be used frequently by the operating system during daily operations.

To do this procedure, execute the `setup` command. For details, see the "System Setup" chapter and `setup`(1M) in the *System Administrator's Reference Manual*.

Step 6: Install Additional Software Packages

Install any software packages that you want to make available to your users, such as the Editing Utilities package. Instructions are in the "Software Management" chapter.

Step 7: Set Up Ports

Enable data connections that can be used to log in on your computer. For instructions, see the "Service Access" chapter.

Step 8: Install Peripheral Devices

Now you are ready to connect terminals and other peripheral devices, such as modems and printers, to your system. These connections are made through outlets on your computer called I/O (input/output) ports. Before you can use a port, you must allocate it for use by a particular device. Instructions for doing this are in the "Service Access" chapter of this book. Once you have allocated the ports on your system, connect your terminals, printers, and modems. For details about installing printers, see the manual for your printer and the "Print Service" chapter of this book.

Install Data Storage Devices

One important category of peripheral devices is storage devices, such as disk drives and cartridge tape drives (also known as block devices and character devices, respectively). These devices allow users to record data on removable storage media, such as floppy diskettes and cartridge tapes. To learn how to install storage devices, see the "Storage Device Management" chapter of this book. (The latter chapter also includes instructions for formatting, copying, and using—as mountable file systems—removable storage media.)

Install the LP Print Service

If you want to make printers available to your users, you should now install the printers and the LP print service software. See your printer installation manual for hardware installation instructions, and the "Software Management" and "Print Service" chapters for software installation instructions.

Step 9: Customize the System Profile

Your system is delivered with a default system profile (`/etc/profile`) that defines the basic operating environment for the users on your system. If you want to change any of the parameters of this profile, see the instructions in the "User and Group Management" chapter.

Step 10: Create Groups for Users

Users frequently want to share data but allowing them to do so without restrictions is unadvisable because system security may thus be compromised. To protect data while allowing those users who need to share data to do so, the system allows you to assign users to groups. Once a user has been assigned to one or more groups, the members of those groups are automatically granted permission to use the data in any directories or files created by that user.

To make sure that a group assignment is available for every new user account on your system, even if you do not create groups, UNIX System V provides a file called `/etc/group`. If you do not create groups before assigning user login names and passwords, all new users will be assigned to the group `other` by default.

If you want to create the user groups for your system, see the "User and Group Management" chapter for instructions.

Step 11: Assign User Logins and Passwords

Now you are ready to start letting people use the system; you need to create user login names and passwords. There are two reasons for doing this. First, as a security measure, UNIX System V will not allow anyone to log in on your system without a login name and a password.

Second, by creating a login name for someone, you are also giving that person an account on your system. Ownership of an account affords not only access to the computer's resources, but also a working environment defined by the system profile and user profile you provide.

To find out how to assign user login names and passwords, see the "User and Group Management" chapter.

Step 12: Set Up a Network

If you want to enable your users to communicate with users on other computers, and to use the resources (such as printers) available on those computers, you can connect your system to others through a network. See the *Network User's and Administrator's Guide* and the "Network Services" chapter of this book.

Allocation of System Resources

The job of allocating resources implies two tasks: providing and adjusting those resources and overseeing access to them. Both these tasks must be done on an ongoing basis, as the need for them arises.

The first task is to provide resources. Specifically, you can do the following:

- Give users more space in which to work by creating and mounting new file systems. For instructions, see the "File System Administration" chapter.
- Give users additional space for storing data by adding peripheral devices such as disk drives, floppy diskette drives, and cartridge tape drives. For instructions, see the manual for the device you are installing.
- Install software packages. For instructions, see the "Software Management" chapter.

Your second responsibility for resource allocation is controlling who has access to your computer. You can do this by assigning login names and passwords to only those authorized by you to use your system. This task is one of the administrator's first jobs when setting up a system, as described under "Step 11: Assign User Logins and Passwords" in the previous section. Unlike other initial setup tasks, however, assigning user login names and passwords is a job you will have to do from time to time, as personnel changes in your workplace demand. For instructions on assigning logins and passwords, see the "User and Group Management" chapter.

For tips about how to keep your system secure by using login names and passwords judiciously, see the "Security" chapter.

The UNIX System V Release 4.0 process scheduler is tuned to provide good performance over a wide range of computing environments. If you have special requirements for process scheduling, however, the scheduler offers great flexibility. For time-sharing processes, you can control how the scheduler assigns priorities and time slices to user processes. You can also configure real-time processes, which allow privileged users direct control over the order in which processes run. See the "Process Scheduling" chapter for details.

Optimized Use of Software Resources

To make the most efficient use of your system, it's a good idea to track, at regular intervals, how resources are being used and how well your system performs in response to users' requests. If your analysis reveals the system is not running as efficiently as possible, you may want to implement performance improvement measures.

In addition, there may be times when the system response slows down noticeably. If this happens, you will need to investigate possible reasons for performance degradation. Such slowdowns are generally caused by either memory bottlenecks or I/O bottlenecks.

UNIX System V provides a set of tools called the System Performance Analysis Utilities (or SPAU) which, when used with other basic UNIX system utilities, allow you to detect possible performance problems.

These utilities can be classified broadly into two sets. One set of utilities reports information about system activity that is being recorded continuously in the operating system kernel. These tools can be used to do the following:

- collect system activity data, automatically and on demand
- display system activity reports on activities such as CPU utilization, paging, buffering, disk activity, queue activity, and tty activity
- time a command and look at system activity during the execution of that command

A second set of utilities reports statistics about disk and file system usage that are gathered only by request. These tools can be used to do the following:

- report disk access location and seek distance
- track files based on size or age
- print the number of free file blocks and inodes
- summarize file system usage

With the information gathered with these tools, you may decide to change the configuration of your system in ways that can improve the response time and overall efficiency of your system. For instructions on using the SPAU utilities, see the "Performance Management" chapter.

Another set of tools, the Accounting Utilities, allows you to check system usage by providing answers to questions such as the following:

- Who is using which resources?
- What patterns of command usage can be identified? For example, which commands are most frequently used?
- How much disk space is being used by which users?

The Accounting Utilities can also be used to calculate billing charges for use of computer resources.

For instructions on using these utilities, see the "Accounting" chapter.

Protection of System Resources

As the system administrator, you are responsible for protecting the data and software on a computer. Here are a few procedures that can help you do this.

- Identify administrative and system functions that should be done only by the administrator (or an authorized delegate) and assign passwords to these tasks. You can do this during the initial setup procedure described in the "System Setup" chapter. You can always change or add to the passwords assigned during that procedure, however. For instructions, see the "User and Group Management" chapter.
- Set up a regular schedule for backing up (copying) the data on your system. Decide how often you must back up various data objects (full file systems, partial file systems, data partitions, and so on) to ensure that lost data can always be retrieved. See the "Backup Service" and "Restore Service" chapters for instructions.
- Control the permissions codes (which restrict access) for important administrative directories and files. You can do this during the initial setup procedure described in the "System Setup" chapter. You can always change or add to the permissions assigned during that procedure, however. For instructions, see the "User and Group Management" chapter.

For general guidelines about how to make sure your system is secure from intrusion, data corruption, and data loss, see the "Security" chapter.

Routine Maintenance

Finally, from time to time you will need to do certain administrative tasks to ensure that your system continues to function in a healthy way.

- Do backups of your system at regularly scheduled intervals. See the "Backup Service" chapter for a discussion of the various types of backups that can be done and suggestions about how often each type should be done.
- Make sure your software is up to date. When new releases of software used on your system become available, install them to provide your users with the best possible tools. For installation instructions, see the "Software Management" chapter.
- Clean the heads on your floppy disk drives and your tape drives regularly.
- Monitor the space available on each of your file systems. If the available space on a file system gets too low, you will have to take some action to increase the space. The possibilities include moving files from a full file system to a file system with more space, emptying or truncating system log files, and asking users to delete unnecessary files. See "Maintaining a File System" in the "File System Administration" chapter.

Some of these tasks can be done only when your computer is operating in firmware state or after it has been powered down. Thus you will need to change the system state of your computer on a regular basis. To save time, you may want to define a default program capable of booting the system, powering down and rebooting the system, and entering firmware state. The "Machine Management" chapter of this book explains how to create such a program.

Repairs of Defective Hardware and Software

An important function of a system administrator is to identify and fix problems that occur in both the hardware and software while the system is in normal use. The UNIX system provides a set of tools that allows you to pinpoint hardware malfunctions. These tools, along with a few troubleshooting suggestions, are described in the "Diagnostics" chapter. This chapter also explains how to handle bad blocks on the hard disk.

Software related problems are handled separately. As the administrator of your system, you must become familiar with the file system so that you can do the following:

- investigate and repair software related errors on a specific file system
- monitor disk usage for all file systems
- track files based on age or size
- create new file systems
- mount and unmount file systems

See the "File System Administration" chapter for instructions.

Guidelines for Good Customer Service

As a system administrator, you are responsible for providing the best possible service to your customers, the users on your system.

Maintaining a System Log

If your system supports multiple users, we strongly recommend that you keep a record of system activities in a log. A system log book can be a valuable tool when troubleshooting transient problems or when trying to identify patterns in the way your system operates and is used. Therefore we recommend that you record any information that may prove useful later including, at least, the following:

- dates and descriptions of maintenance procedures
- printouts of error messages and diagnostic phases
- dates and descriptions of hardware changes
- dates and descriptions of software changes

Keeping Users Informed About Administrative Issues

Users need to know when a computer will be taken out of service, when a new software package will become available, and who to call for help when they encounter a problem with the system. To keep users informed about changes in hardware, software, administrative policies, and procedures, send them electronic messages with one of the communications tools provided with UNIX System V. Use the message of the day file (`/etc/motd`) for sending daily reminders and announcements. (For details, see the "User and Group Management" chapter.) For distributing information on an ad hoc basis, use the news facility (see `news(1)` for details).

Shutting Down the System

Many administrative tasks require the system to be shut down to a system state other than multi-user state (see the "System States" section in the "Machine Management" chapter). While the computer is not in multi-user state, your customers cannot access it. As a courtesy to these users, follow the suggestions below when you're planning administrative work that may disrupt their daily activities.

- Schedule tasks that will affect service for periods of low system use.
- Before taking any actions that might affect a user who is logged on, check to see who is on the system by running the whodo command (see whodo(1M)) or the who command (see who(1)).
- If the system is in use, give users advance warning about pending changes in system states or maintenance actions by running the `wall` command (see `wall`(1M)). Give users a reasonable amount of time to finish their activities and log off before taking the system down. If possible, tell users when they can expect the system to return to service.
- When unscheduled servicing occurs, run the `wall` command (see `wall`(1M)) to notify users. Again, if possible, tell them when they can expect the system to return to service.

2 Accounting

Introduction

The UNIX system accounting utilities are a family of mechanisms that collect data on system usage by CPU usage, by user, and by process. The utilities include tools for keeping track of connect sessions and disk usage. The accounting utilities can be used for

- charging for usage
- troubleshooting performance problems
- tuning the performance of applications
- managing installation security

To help you access the data captured by these utilities, the accounting utilities provide C language programs and shell scripts that organize the data into summary files and reports.

This chapter describes how the accounting utilities work. Specifically, it describes the numerous files and programs that figure prominently in the accounting system. The chapter also provides samples of the various reports generated by the accounting utilities.

Overview of Accounting

Once it has been set up, UNIX system accounting runs mostly on its own. (For instructions on setting up an accounting system, see "Setting Up Accounting" later in this chapter.) The following is an overview of how accounting works.

- Between system start-up and shutdown, raw data about system use (such as logins, processes run, and data storage) are collected in accounting files.
- Once a day, `cron` invokes the `runacct` program, which processes the various accounting files and produces both cumulative summary files and daily accounting reports. The daily reports are then printed by the `prdaily` program, which is invoked by `runacct`.
- The cumulative summary files generated by `runacct` can be processed and printed monthly by executing the `monacct` program. The summary reports produced by `monacct` provide an efficient means for billing users on a monthly or other fiscal basis.

Types of Accounting

The data collected daily with the procedure described above can help you do four types of accounting: connect accounting, process accounting, disk accounting, and fee calculations. The rest of this introduction defines these four types of accounting.

Connect Accounting

Connect accounting enables you to determine how long a user was logged in, and to obtain information about the usage of tty lines, the number of reboots on your system, and the frequency of the stopping and starting of the accounting software. To provide this information, the system stores records of time adjustments, boot times, the turning on or off of the accounting software, changes in run levels, the creation of user processes, login processes, and `init` processes, and the deaths of processes. These records (produced from the output of system programs such as `date`, `init`, `login`, `ttymon`, and `acctwtmp`) are stored in `/var/adm/wtmp`. Entries in the `wtmp` directory may contain the following information: a user's login name, a device name, a process ID, the type of entry, and a time stamp denoting when the entry was made.

Process Accounting

Process accounting allows you to keep track of the following data about each process run on your system: the user and group IDs of those using the process, the beginning and elapsed times of the process, the CPU time for the process (divided between users and the system), the amount of memory used, the commands run, and the controlling tty during the process. Every time a process dies, the `exit` program collects these data and writes them to the file `/var/adm/pacct`.

The `pacct` file has a default maximum size of 500 blocks that is enforced by the accounting shell script `ckpacct` (normally run as a `cron` job). If `ckpacct` finds that `/var/adm/pacct` is over 500 blocks, it moves the file to `/var/adm/pacct`*?* where *?* is the next unused increment (expressed as a number).

Disk Accounting

Disk accounting allows you to gather (and format) the following data about the files each user has on disks: the name and ID of the user, and the number of blocks used by the user's files. These data are collected by four programs in the accounting package: a shell script called `dodisk` and three C programs that it invokes (`diskusg`, `acctdusg`, and `acctdisk`). `diskusg` gathers the file data by reading file inodes directly from the file system and works only for s5 file systems. `acctdusg` does stat calls for each file in the file system tree to gather data and works for any file system type. `diskusg` is faster than `acctdusg`. `acctdisk` formats the data gathered by `diskusg` and/or `acctdusg` and saves the information in `/var/adm/acct/nite/disktacct`.

The `dodisk` script can be used in either of two ways: in fast mode or in slow mode. Fast mode uses `diskusg` for all s5 file systems and `acctdusg` for all others. The fast mode syntax is:

`/usr/lib/acct/dodisk` *file_systems*

File systems are specified by their special device names (such as `/dev/dsk/c1d0s2`). If the file systems are not specified, then the file systems used are those found in `/etc/vfstab` for which the value of `fsckpass` is `1`.

When run in slow mode, `dodisk` invokes `acctdusg` to gather all the disk accounting information, even from s5 file systems. The slow mode syntax is:

`/usr/lib/acct/dodisk -o` *mountpoints*

If no mountpoints are specified, the root mountpoint is used.

One note of caution: information gathered by running `dodisk` in either fast mode or slow mode is stored in the `/var/adm/acct/nite/disktacct` file. This information is overwritten the next time `dodisk` is used. Therefore, avoid using both modes on the same day. This allows `runacct` to use the information in the `/var/adm/acct/nite/disktacct` file before it is overwritten by new output from `dodisk`.

NOTE diskusg may overcharge for files written in random access fashion, that have created holes in the file. This is because diskusg does not read the indirect blocks of a file when determining its size. Rather, diskusg determines the size of a file by looking at the *di_size* value of the inode.

Fee Calculations

If you charge your users for special services, such as file restores and remote printing, you may want to use a program called chargefee to maintain service accounts. Fees charged to customers are recorded in a file called /var/adm/fee. Each entry in the file consists of a user's login name, user ID, and the fee.

Accounting Programs

All the C language programs and shell scripts necessary to run the accounting system are in the /usr/src/cmd/acct directory. The acctcom program is stored in /usr/bin; all other binary programs are stored in /usr/lib/acct. These programs, which are owned by bin (except accton, which is owned by root), do various functions. For example, /usr/lib/acct/startup helps initiate the accounting process when the system enters multi-user mode. The chargefee program is used to charge a particular user for a special service, such as performing a file restore from tape. Other essential programs in the /usr/lib/acct directory include monacct, prdaily, and runacct. These and other programs are discussed in more detail in the following sections.

Setting Up Accounting

To set up system accounting so that it will be running while the system is in multi-user mode (system state 2), four files need to be created and/or modified. These files are /sbin/rc0.d/K22acct, /var/spool/cron/crontabs/adm, /sbin/rc2.d/S22acct, and /var/spool/cron/crontabs/root.

If you want accounting to be shut off during shutdown, link /sbin/init.d/acct to etc/rc0.d/k22acct.

If you want accounting to be turned on when the system is in multi-user mode (system state 2), link /sbin/init.d/acct to /sbin/rc2.d/S22acct.

Most of the cron entries needed for accounting are put into a database called /var/spool/cron/crontabs/adm. The entries in this database allow ckpacct to be run periodically, runacct to be run daily, and monacct to be run on a fiscal basis. Figure 2-1 shows several sample entries; your entries may vary. Be sure to append this information to the file to avoid destroying any entries already present. For the adm crontab, assign root as the owner, sys as the group, and 644 as the permissions mode.

Figure 2-1: Sample cron Entries for Accounting

```
----------------entries for adm crontab-------------------------

#Min  Hour  Day      Month    Day   Command
#           of                of
#           Month             Week
#---------------------------------------------------------------
0     *     *        *        *     /usr/lib/acct/ckpacct
30    2     *        *        *     /usr/lib/acct/runacct 2> /var/adm/acct/nite/fd2log
30    9     *        *        5     /usr/lib/acct/monacct
----------------------------------------------------------------
```

The entry for dodisk needs to be appended to the root crontab /var/spool/cron/crontabs/root. A sample is shown below.

```
------------------entry for root crontab-----------------------------

#Min  Hour  Day       Month     Day   Command
#           of                  of
#           Month               Week
#---------------------------------------------------------------------
30    22    *         *         4     /usr/lib/acct/dodisk
```

Once these entries are in the database and the accounting programs have been installed, accounting will pretty much run on its own.

Daily Accounting

Here is a step-by-step summary of how UNIX system accounting works:

1. When the UNIX system is switched into multi-user mode, the `/usr/lib/acct/startup` program is executed. The `startup` program executes several other programs that invoke accounting:

   - The `acctwtmp` program adds a "boot" record to `/var/adm/wtmp`. In this record the system name is shown as the login name in the `wtmp` record. Figure 2-2 presents a summary of how the raw accounting data is gathered and where it is stored.

Figure 2-2: Raw Accounting Data

File	Information	Written By	Format
`/var/adm/wtmp`	connect sessions	`login`, `init`	`utmp.h`
	date changes	`date`	
	reboots	`acctwtmp`	
	shutdowns	`shutacct` shell	
`/var/adm/pacct?`	processes	kernel (when process ends)	`acct.h`
		`turnacct` switch creates new file when old one reaches 500 blocks	
`/var/adm/fee`	special charges	`chargefee`	
`/var/adm/acct/nite/disktacct`	disk space used	`dodisk`	`tacct.h`

   - The `turnacct` program, invoked with the on option, begins process accounting. Specifically, `turnacct on` executes the `accton` program with the argument `/var/adm/pacct`.

   - The `remove` shell script "cleans up" the saved `pacct` and `wtmp` files left in the `sum` directory by `runacct`.

2. The login and init programs record connect sessions by writing records into /var/adm/wtmp. Any date changes (using date with an argument) are also written to /var/adm/wtmp. Reboots and shutdowns (via acctwtmp) are also recorded in /var/adm/wtmp.

   When a process ends, the kernel writes one record per process, in the form of acct.h, in the /var/adm/pacct file.

   Two programs track disk usage by login: acctdusg and diskusg. They are invoked by the shell script dodisk.

   Every hour cron executes the ckpacct program to check the size of /var/adm/pacct. If the file grows past 500 blocks (default), turnacct switch is executed. (The turnacct switch program moves the pacct file and creates a new one.) The advantage of having several smaller pacct files becomes apparent when trying to restart runacct if a failure occurs when processing these records.

   If the system is shut down using shutdown, the shutacct program is executed automatically. The shutacct program writes a reason record into /var/adm/wtmp and turns off process accounting.

   If you provide services on a request basis (such as file restores), you can keep billing records by login, by using the chargefee program. It allows you to add a record to /var/adm/fee each time a user incurs a charge. The next time runacct is executed, this new record is picked up and merged into the total accounting records.

3. runacct is executed via cron each night. It processes the accounting files /var/adm/pacct?, /var/adm/wtmp, /var/adm/fee, and /var/adm/acct/nite/disktacct to produce command summaries and usage summaries by login.

4. The /usr/lib/acct/prdaily program is executed on a daily basis by runacct to write the daily accounting information collected by runacct (in ASCII format) in /var/adm/acct/sum/rprt*MMDD*.

5. The monacct program should be executed on a monthly basis (or at intervals determined by you, such as the end of every fiscal period). The monacct program creates a report based on data stored in the sum directory that has been updated daily by runacct. After creating the report,

monacct "cleans up" the sum directory to prepare the directory's files for the new runacct data.

The runacct Program

The main daily accounting shell procedure, runacct, is normally invoked by cron during non-prime time hours. The runacct shell script processes connect, fee, disk, and process accounting files. It also prepares daily and cumulative summary files for use by prdaily and monacct for billing purposes.

The runacct shell script takes care not to damage files if errors occur. A series of protection mechanisms are used that attempt to recognize an error, provide intelligent diagnostics, and end processing in such a way that runacct can be restarted with minimal intervention. It records its progress by writing descriptive messages into the file active. (Files used by runacct are assumed to be in the /var/adm/acct/nite directory unless otherwise noted.) All diagnostic output during the execution of runacct is written into fd2log.

If the files lock and lock1 exist when invoked, runacct will complain. These files are used to prevent simultaneous execution of runacct. The lastdate file contains the month and day runacct was last invoked and is used to prevent more than one execution per day. If runacct detects an error, a message is written to the console, mail is sent to root and adm, locks are removed, diagnostic files are saved, and execution is ended.

Reentrant States of the runacct Script

To allow runacct to be restartable, processing is broken down into separate reentrant states. A file is used to remember the last state completed. When each state completes, statefile is updated to reflect the next state. After processing for the state is complete, statefile is read and the next state is processed. When runacct reaches the CLEANUP state, it removes the locks and ends. States are executed as follows:

SETUP The command turnacct switch is executed to create a new pacct file. The process accounting files in /var/adm/pacct? (except the pacct file) are moved to /var/adm/Spacct?.*MMDD*. The /var/adm/wtmp file is moved to /var/adm/acct/nite/wtmp.*MMDD* (with the current time record added on the end) and a new /var/adm/wtmp is created. closewtmp and utmp2wtmp add records to wtmp.*MMDD* and the new wtmp to account for users currently logged in.

WTMPFIX	The wtmpfix program checks the wtmp.*MMDD* file in the nite directory for correctness. Because some date changes will cause acctcon to fail, wtmpfix attempts to adjust the time stamps in the wtmp file if a record of a date change appears. It also deletes any corrupted entries from the wtmp file. The fixed version of wtmp.*MMDD* is written to tmpwtmp.
CONNECT	The acctcon program is used to record connect accounting records in the file ctacct.*MMDD*. These records are in tacct.h format. In addition, acctcon creates the lineuse and reboots files. The reboots file records all the boot records found in the wtmp file. CONNECT was previously two steps called CONNECT1 and CONNECT2.
PROCESS	The acctprc program is used to convert the process accounting files /var/adm/Spacct?.*MMDD*, into total accounting records in ptacct?.*MMDD*. The Spacct and ptacct files are correlated by number so that if runacct fails, the unnecessary reprocessing of Spacct files will not occur. One precaution should be noted: when restarting runacct in this state, remove the last ptacct file because it will not be complete.
MERGE	Merge the process accounting records with the connect accounting records to form daytacct.
FEES	Merge in any ASCII tacct records from the file fee into daytacct.
DISK	If the dodisk procedure has been run, producing the file disktacct, merge the file into daytacct and move disktacct to /tmp/disktacct.*MMDD*.
MERGETACCT	Merge daytacct with sum/tacct, the cumulative total accounting file. Each day, daytacct is saved in sum/tacct.*MMDD*, so that sum/tacct can be recreated if it is corrupted or lost.
CMS	The program acctcms is run several times. It is first run to generate the command summary using the Spacct? files and writes it to sum/daycms. acctcms is then run to merge sum/daycms with the cumulative command summary file sum/cms. Finally, acctcms is run to produce the ASCII command summary files nite/daycms and nite/cms from the files sum/daycms and sum/cms,

respectively. The program lastlogin is used to create /var/adm/acct/sum/loginlog, the report of when each user last logged on. (If runacct is run after midnight, the dates showing the time last logged on by some users will be incorrect by one day.)

USEREXIT

Any installation-dependent (local) accounting program can be included here. runacct expects it to be called /usr/lib/acct/runacct.local.

CLEANUP Clean up temporary files, run prdaily and save its output in sum/rprt*MMDD*, remove the locks, then exit.

runacct Error Messages

The runacct procedure can fail for a variety of reasons; the most frequent reasons are a system crash, /var running out of space, and a corrupted wtmp file. If the active*MMDD* file exists, check it first for error messages. If the active file and lock files exist, check fd2log for any mysterious messages. The following are error messages produced by runacct and the recommended recovery actions:

```
ERROR: locks found, run aborted
```

The files lock and lock1 were found. These files must be removed before runacct can restart. Either two processes are trying to run runacct simultaneously or the last runacct aborted abnormally without cleaning up the locks. Check the fd2log for messages.

```
ERROR: acctg already run for date: check
       /var/adm/acct/nite/lastdate
```

The date in lastdate and today's date are the same. Remove lastdate.

```
ERROR: turnacct switch returned rc=?
```

Check the integrity of turnacct and accton. The accton program must be owned by root and have the setuid bit set.

ERROR: Spacct?.*MMDD* already exists

File setups probably already run. Check status of files, then run setups manually, if necessary.

ERROR: /var/adm/acct/nite/wtmp.*MMDD* already exists, run setup manually

/var/adm/wtmp has already been copied to /var/adm/acct/nite/wtmp.*MMDD*

ERROR: wtmpfix errors see /var/adm/acct/nite/wtmperror

wtmpfix detected a corrupted wtmp file. Use fwtmp to correct the corrupted file.

ERROR: Invalid state, check /var/adm/acct/nite/statefile

The file statefile is probably corrupted. Check statefile and read active before restarting.

Files Produced by runacct

The following files produced by runacct (found in /var/adm/acct) are of particular interest:

nite/lineuse	runacct calls acctcon to gather data on terminal line usage from /var/adm/acct/nite/tmpwtmp and writes the data to /var/adm/acct/nite/lineuse. prdaily uses this data to report line usage. This report is especially useful for detecting bad lines. If the ratio between the number of logoffs to logins exceeds about 3:1, there is a good possibility that the line is failing.
nite/daytacct	This file is the total accounting file for the day in tacct.h format.
sum/tacct	This file is the accumulation of each day's nite/daytacct and can be used for billing purposes. It is restarted each month or fiscal period by the monacct procedure.

sum/daycms	runacct calls acctcms to process the data about the commands used during the day. This information is stored in /var/adm/acct/sum/daycms. It contains the daily command summary. The ASCII version of this file is /var/adm/acct/nite/daycms.
sum/cms	This file is the accumulation of each day's command summaries. It is restarted by the execution of monacct. The ASCII version is nite/cms.
sum/loginlog	runacct calls lastlogin to update the last date logged in for the logins in /var/adm/acct/sum/loginlog. lastlogin also removes from this file logins that are no longer valid.
sum/rprt*MMDD*	Each execution of runacct saves a copy of the daily report that was printed by prdaily.

Fixing Corrupted Files

Unfortunately, this accounting system is not entirely foolproof. Occasionally, a file will become corrupted or lost. Some of the files can simply be ignored or restored from the backup. However, certain files must be fixed to maintain the integrity of the accounting system.

Fixing `wtmp` Errors

The `wtmp` files seem to cause the most problems in the day-to-day operation of the accounting system. When the date is changed and the system is in multi-user mode, a set of date change records is written into `/var/adm/wtmp`. The `wtmpfix` program is designed to adjust the time stamps in the `wtmp` records when a date change is encountered. However, some combinations of date changes and reboots will slip through `wtmpfix` and cause `acctcon` to fail. The following steps show how to patch up a `wtmp` file.

Figure 2-3: Repairing a `wtmp` File

```
cd /var/adm/acct/nite
fwtmp < wtmp.MMDD > xwtmp
ed xwtmp
   delete corrupted records or
   delete all records from beginning
   up to the date change
w
q
fwtmp -ic < xwtmp > wtmp.MMDD
```

If the `wtmp` file is beyond repair, create a null `wtmp` file. This will prevent any charging of connect time. As a side effect, the lack of a `wtmp` file prevents `acctprc` from identifying the login that owned a particular process; the process is charged to the owner of the first login in the password file for the appropriate user ID.

Fixing tacct Errors

If the installation is using the accounting system to charge users for system resources, the integrity of `sum/tacct` is important. Occasionally, mysterious `tacct` records will appear with negative numbers, duplicate user IDs, or a user ID of `65,535`. First, check `sum/tacctprev`, using `prtacct` to print it. If it looks all right, the latest `sum/tacct`.*MMDD* should be patched up, then `sum/tacct` recreated. A simple patchup procedure would be:

Figure 2-4: Repairing a tacct File

```
cd /var/adm/acct/sum
acctmerg -v < tacct.MMDD > xtacct
ed xtacct
    remove the bad records
    write duplicate uid records to another file
w
q
acctmerg -i < xtacct > tacct.MMDD
acctmerg tacctprev < tacct.MMDD > tacct
```

The current `sum/tacct` can be recreated by merging all existing `tacct`.*MMDD* files by using `acctmerg`, since the `monacct` procedure removes all the old `tacct`.*MMDD* files.

Restarting `runacct`

Called without arguments, `runacct` assumes that this is the first invocation of the day. The argument *MMDD* is necessary if `runacct` is being restarted and specifies the month and day for which `runacct` will rerun the accounting. The entry point for processing is based on the contents of `statefile`. To override `statefile`, include the desired state on the command line. The following are some sample procedures.

To start `runacct`:

```
nohup runacct  2>  /var/adm/acct/nite/fd2log &
```

To restart `runacct`:

```
nohup runacct 0601  2>  /var/adm/acct/nite/fd2log &
```

To restart `runacct` in a specific state:

```
nohup runacct 0601 WTMPFIX  2>  /var/adm/acct/nite/fd2log &
```

Billing Users

The `chargefee` program stores charges for special services provided to a user, such as file restores, in the file `fee`. This file is incorporated by `runacct` every day.

To register special fees, enter the following command:

`chargefee` *login_name amount*

where *amount* is an integer amount to be charged. Most locations prefer to set up their own shell scripts for this function, with codes for services rendered. The operator then need identify only the service rendered; the system can tabulate the charge.

The monthly accounting program `monacct` produces monthly summary reports similar to those produced daily. (See Figure 2-9 later in this chapter for a sample report.) The `monacct` program also summarizes the accounting information into the files in the `/var/adm/acct/fiscal` directory. This information can be used to generate monthly billing. To generate a monthly billing, many UNIX system administrators customize the accounting process with their own shell scripts.

Setting Up Non-Prime Time Discounts

UNIX system accounting provides facilities to give users a discount for non-prime time system use. For this to work, you must inform the accounting system of the dates of holidays and the hours that are considered non-prime time, such as weekends. To do this, you must edit the `/etc/acct/holidays` file that contains the prime/non-prime table for the accounting system. The format is composed of three types of entries:

- Comment Lines—Comment lines are marked by an asterisk in the first column of the line. Comment lines may appear anywhere in the file.
- Year Designation Line—This line should be the first data line (noncomment line) in the file and must appear only once. The line consists of three fields of four digits each (leading white space is ignored). For example, to specify the year as 1990, prime time start at 9:00 A.M. , and non-prime time start at 4:30 P.M. , the following entry would be appropriate:

```
1990  0900   1630
```

A special condition allowed for in the time field is that the time 2400 is automatically converted to 0000.

- Company Holidays Lines—These entries follow the year designation line and have the following general format:

  Date   Description of Holiday

  The date field has the format *month/day* and indicates the date of the holiday. The holiday field is actually commentary and is not currently used by other programs. See Figure 2-5 for an example holiday list.

Figure 2-5: Holiday List

Month/Day	Holiday
1/1	New Year's Day
5/28	Memorial Day
7/4	Independence Day
9/3	Labor Day
11/22	Thanksgiving Day
11/23	Day after Thanksgiving
12/25	Christmas Day

Daily Accounting Reports

The runacct shell script generates four basic reports upon each invocation. They cover the areas of connect accounting, usage by login on a daily basis, command usage reported by daily and monthly totals, and a report of the last time users were logged in. The four basic reports generated are:

- The daily report shows line utilization by tty number.
- The daily usage report indicates usage of system resources by users (listed in order of UID).
- The daily command summary indicates usage of system resources by commands, listed in descending order of use of memory (in other words, the command that used the most memory is listed first). This same information is reported for the month with the monthly command summary.
- The last login shows the last time each user logged in (arranged in chronological order).

The following paragraphs describe the reports and the meaning of the data presented in each one.

Daily Report

This report gives information about each terminal line used. Figure 2-6 shows a sample daily report.

Figure 2-6: Sample Daily Report

```
Jun 29 09:53 1990  DAILY REPORT FOR sfxbs Page 1


from Thu Jun 28 17:45:22 1990
to   Fri Jun 29 09:51:25 1990
1         runacct
1         acctcon

TOTAL DURATION IS 966 MINUTES
LINE      MINUTES   PERCENT   # SESS    # ON      # OFF
term/23             25        3         7         4         4
term/22             157       16        6         3         3
TOTALS              183       --        13        7         7

-----------------------------------------------------------------
```

The `from` and `to` lines tell you the time period reflected in the report: the period from the time the last accounting report was generated until the time the current accounting report was generated. It is followed by a log of system reboots, shutdowns, power fail recoveries, and any other record dumped into `/var/adm/wtmp` by the `acctwtmp` program; see `acct`(1M) in the *System Administrator's Reference Manual*.

The second part of the report is a breakdown of line utilization. The `TOTAL DURATION` tells how long the system was in multi-user state (accessible through the terminal lines). The columns are:

`LINE:`	the terminal line or access port
`MINUTES:`	the total number of minutes that line was in use during the accounting period
`PERCENT:`	the total number of `MINUTES` the line was in use, divided into the `TOTAL DURATION`
`# SESS:`	the number of times this port was accessed for a `login` session

`# ON:`	This column does not have much meaning anymore. It used to list the number of times that a port was used to log a user on; but because `login` can no longer be executed explicitly to log in a new user, this column should be identical with SESS.
`# OFF:`	This column reflects not just the number of times a user logs off but also any interrupts that occur on that line. Generally, interrupts occur on a port when `ttymon` is first invoked when the system is brought to multi-user state. Where this column does come into play is when the `# OFF` exceeds the `# ON` by a large factor. This usually indicates that the multiplexer, modem, or cable is going bad, or there is a bad connection somewhere. The most common cause of this is an unconnected cable dangling from the multiplexer.

During real time, you should monitor `/var/adm/wtmp` because it is the file from which the connect accounting is geared. If the `wtmp` file grows rapidly, execute `acctcon -l` *file* `< /var/adm/wtmp` to see which tty line is the noisiest. If the interrupting is occurring at a furious rate, general system performance will be affected.

Daily Usage Report

The daily usage report gives a breakdown of system resource utilization by user. Figure 2-7 shows a sample of this type of report.

Figure 2-7: Sample Daily Usage Report

```
Jun 29 09:53 1990  DAILY USAGE REPORT FOR sfxbs Page 1

     LOGIN  CPU (MINS)   KCORE-MINS    CONNECT (MINS)  DISK    # OF  # OF  # DISK   FEE
UID  NAME   PRIME NPRIME PRIME NPRIME  PRIME  NPRIME   BLOCKS  PROCS SESS  SAMPLES
0    TOTAL  5     12     6     16      131    51       0       1114  13    0        0
0    root   2      8     1     11        0     0       0        519   0    0        0
3    sys    0      1     0      1        0     0       0         45   0    0        0
4    adm    0      2     0      1        0     0       0        213   0    0        0
5    uucp   0      0     0      0        0     0       0         53   0    0        0
999  rly    3      1     5      2      111    37       0        269   1    0        0
7987 jan    0      0     0      1       20    14       0         15   6    0        0
```

The data provided include the following:

`UID:`	The user ID
`LOGIN NAME:`	The login name of the user. This information is useful because it identifies a user who has multiple login names.
`CPU (MINS):`	This represents the amount of time the user's process used the central processing unit. This category is broken down into `PRIME` and `NPRIME` (non-prime) utilization. The accounting system's idea of this breakdown is located in the `/etc/acct/holidays` file.
`KCORE-MINS:`	This represents a cumulative measure of the amount of memory a process uses while running. The amount shown reflects kilobyte segments of memory used per minute. This measurement is also broken down into `PRIME` and `NPRIME` amounts.

`CONNECT` and `(MINS):`

This identifies the amount of "real time" used. What this column really identifies is the amount of time that a user was logged into the system. If the amount of time is high and the number shown in the column `# OF PROCS` is low, you can safely conclude that the owner of the login for which the report is being generated is a "line hog." That is,

	this person logs in first thing in the morning and hardly touches the terminal the rest of the day. Watch out for this kind of user. This column is also subdivided into `PRIME` and `NPRIME` utilization.
`DISK BLOCKS:`	When the disk accounting programs have been run, the output is merged into the total accounting record (`daytacct`) and shows up in this column. This disk accounting is accomplished by the program `acctdusg`. For accounting purposes, a "block" is 512 bytes.
`# OF PROCS:`	This column reflects the number of processes that were invoked by the user. This is a good column to watch for large numbers indicating that a user may have a shell procedure that has run out of control.
`# OF SESS:`	The number of times a user logged on to the system is shown in this column.
`# DISK SAMPLES:`	This indicates how many times the disk accounting was run to obtain the average number of `DISK BLOCKS` listed earlier.
`FEE:`	An often unused field in the total accounting record, the `FEE` field represents the total accumulation of widgets charged against the user by the `chargefee` shell procedure; see `acctsh`(1M). The `chargefee` procedure is used to levy charges against a user for special services performed such as file restores.

Daily Command Summary

The daily command summary report shows the system resource utilization by command. With this report, you can identify the most heavily used commands and, based on how those commands use system resources, gain insight on how best to tune the system. The daily command and monthly reports are virtually the same except that the daily command summary reports only on the current accounting period while the monthly total command summary tells the story for the start of the fiscal period to the current date. In other words, the monthly report reflects the data accumulated since the last invocation of monacct.

These reports are sorted by TOTAL KCOREMIN, which is an arbitrary yardstick but often a good one for calculating "drain" on a system.

Figure 2-8 shows a sample daily command summary.

Figure 2-8: Sample Daily Command Summary

```
Jun 29 09:52 1990  DAILY COMMAND SUMMARY Page 1


                                       TOTAL COMMAND SUMMARY
COMMAND  NUMBER    TOTAL    TOTAL     TOTAL    MEAN     MEAN      HOG    CHARS   BLOCKS
NAME      CMDS   KCOREMIN CPU-MIN  REAL-MIN  SIZE-K  CPU-MIN   FACTOR  TRNSFD    READ

TOTALS    1114      2.44    16.69    136.33    0.15     0.01     0.12  4541666   1926

sh         227      1.01     2.45     54.99    0.41     0.01     0.04   111025    173
fmli        10      0.50     2.06      9.98    0.24     0.21     0.21   182873    223
vi          12      0.35     0.62     44.23    0.55     0.05     0.01   151448     60
sed        143      0.09     0.82      1.48    0.10     0.01     0.55   14505      35
sadc        13      0.08     0.19      1.45    0.44     0.01     0.13   829088     19
more         3      0.04     0.07      2.17    0.59     0.02     0.03    30560      1
cut         14      0.03     0.09      0.28    0.37     0.01     0.33      154     13
uudemon.    76      0.03     0.66      2.30    0.05     0.01     0.29    43661     13
uuxqt       29      0.03     0.30      0.72    0.08     0.01     0.42    80765     35
mail         4      0.02     0.06      0.09    0.37     0.01     0.60     4540      9
ckstr       21      0.02     0.11      0.13    0.17     0.01     0.85        0      4
awk         13      0.02     0.12      0.21    0.15     0.01     0.54      444      2
ps           2      0.02     0.10      0.13    0.17     0.05     0.77     8060     21
find         9      0.02     3.35      5.73    0.00     0.37     0.58   355269    760
sar          1      0.01     0.19      0.24    0.08     0.19     0.80   564224      4
acctdisk     2      0.01     0.01      0.06    1.02     0.01     0.22        0      9
mv          24      0.01     0.14      0.17    0.10     0.01     0.81     3024     36
   .
   .
   .
```

The data provided include the following:

COMMAND NAME	The name of the command. Unfortunately, all shell procedures are lumped together under the name sh because only object modules are reported by the process accounting system. It's a good idea to monitor the frequency of programs called a.out or core or any other name that does not seem quite right. Often people like to work on their favorite version of backgammon, but they do not want everyone to know about it. acctcom is also a good tool to use for determining who executed a suspiciously named command and also if super-user privileges were used.
PRIME NUMBER CMDS	The total number of invocations of this particular command during prime time.
NON-PRIME NUMBER CMDS	The total number of invocations of this particular command during non-prime time.
TOTAL KCOREMIN	The total cumulative measurement of the amount of kilobyte segments of memory used by a process per minute of run time.
PRIME TOTAL CPU-MIN	The total processing time this program has accumulated during prime time.
NON-PRIME TOTAL CPU-MIN	The total processing time this program has accumulated during non-prime time.
PRIME TOTAL REAL-MIN	Total real-time (wall-clock) minutes this program has accumulated.
NON-PRIME TOTAL REAL-MIN	Total real-time (wall-clock) minutes this program has accumulated.
MEAN SIZE-K	This is the mean of the TOTAL KCOREMIN over the number of invocations reflected by NUMBER CMDS.

MEAN CPU-MIN	This is the mean derived between the NUMBER CMDS and TOTAL CPU-MIN.
HOG FACTOR	The total CPU time divided by the elapsed time. This shows the ratio of system availability to system utilization. This gives a relative measure of the total available CPU time consumed by the process during its execution.
CHARS TRNSFD	This column, which may go negative because of overflow, is a total count of the number of characters pushed around by the `read` and `write` system calls.
BLOCKS READ	A total count of the physical block reads and writes that a process performed.

Total Command Summary

The monthly command summary is similar to the daily command summary. The only difference is that the monthly command summary shows totals accumulated since the last invocation of `monacct`. Figure 2-9 shows a sample report.

Figure 2-9: Sample Total Command Summary

```
                                   TOTAL COMMAND SUMMARY

COMMAND NUMBER     TOTAL    TOTAL      TOTAL   MEAN   MEAN     HOG       CHARS    BLOCKS
NAME      CMDS  KCOREMIN  CPU-MIN   REAL-MIN SIZE-K CPUMIN  FACTOR      TRNSFD      READ

TOTALS  301314 300607.70  4301.59  703979.81  69.88   0.01    0.01  6967631360  10596385

troff      480  58171.37   616.15    1551.26  94.41   1.28    0.40   650669248    194926
rnews     5143  29845.12   312.20    1196.93  95.59   0.06    0.26  1722128384   2375741
uucico    2710  16625.01   212.95   52619.21  78.07   0.08    0.00   228750872    475343
nroff     1613  15463.20   206.54     986.06  74.87   0.13    0.21   377563304    277957
vi        3040  14641.63   157.77   14700.13  92.80   0.05    0.01   116621132    206025
expire      14  13424.81   104.90     265.67 127.98   7.49    0.39    76292096    145456
comp      3483  12140.64    60.22     423.54 201.62   0.02    0.14     9584838    372601
ad_d        71  10179.20    50.02    1158.31 203.52   0.70    0.04    11385054     19489
as        2312   9221.59    44.40     285.52 207.68   0.02    0.16    35988945    221113
gone       474   8723.46   219.93   12099.01  39.67   0.46    0.02    10657346     19397
i10        299   8372.60    44.45     454.21 188.34   0.15    0.10    60169932     78664
find       760   8310.97   196.91     728.39  42.21   0.26    0.27    58966910    710074
ld        2288   8232.84    61.19     425.57 134.55   0.03    0.14   228701168    279530
fgrep      832   7585.34    62.62     199.11 121.14   0.08    0.31    22119268     37196
sh       56314   7538.40   337.60  291655.70  22.33   0.01    0.00    93262128    612892
du         624   5049.58   126.32     217.59  39.97   0.20    0.58    16096269    215297
ls       12690   4765.60    75.71     541.53  62.95   0.01    0.14    65759473    207920
vnews       52   4235.71    28.11     959.74 150.70   0.54    0.03    28291679     28285
  .
  .
  .
```

Last Login Report

This report simply gives the date when a particular login was last used. You can use this information to find unused logins and login directories that may be archived and deleted. Figure 2-10 shows a sample report.

Figure 2-10: Sample Last Login

```
Feb 13 04:40 1990  LAST LOGIN Page 1

00-00-00  **RJE** 88-01-01  jlr     88-02-09  cec42   88-02-13   cec20
00-00-00  **rje** 88-01-13  crom    88-02-10  jgd     88-02-13   cec22
00-00-00  3bnet   88-01-14  usg     88-02-10  wbr     88-02-13   cec23
00-00-00  adm     88-01-17  cec11   88-02-11  cec30   88-02-13   cec24
00-00-00  daemon  88-01-17  cec38   88-02-11  cec41   88-02-13   cec25
00-00-00  notes   88-01-17  cec40   88-02-11  cec43   88-02-13   cec26
00-00-00  oas     88-01-18  cec60   88-02-11  cec53   88-02-13   cec27
00-00-00  pds     88-01-19  cec35   88-02-11  cec54   88-02-13   cec3
00-00-00  polaris 88-01-19  cec37   88-02-11  cec55   88-02-13   cec31
00-00-00  rje     88-01-22  dmk     88-02-11  cec56   88-02-13   cec32
00-00-00  shqer   88-01-26  ask     88-02-11  cec57   88-02-13   cec4
00-00-00  sys     88-01-26  cec39   88-02-11  cec58   88-02-13   cec6
00-00-00  trouble 88-01-27  sync    88-02-11  jwg     88-02-13   cec7
00-00-00  usors   88-02-02  pkl     88-02-11  skt     88-02-13   cec8
00-00-00  uucp    88-02-03  ibm     88-02-11  tfm     88-02-13   commlp
00-00-00  wna     88-02-03  slk     88-02-12  cec21   88-02-13   djs
87-07-06  lp      88-02-04  cec59   88-02-12  cec28   88-02-13   epic
87-07-30  dgn     88-02-05  cec33   88-02-12  cec29   88-02-13   jab
87-08-19  blg     88-02-05  cec34   88-02-12  csp     88-02-13   jcs
87-12-08  emna    88-02-05  cec36   88-02-12  drc     88-02-13   mak
88-01-14  s       88-02-05  cec51   88-02-12  emw     88-02-13   mdn
88-01-09  rib     88-02-05  dfh     88-02-12  je      88-02-13   mlp
88-01-25  dmf     88-02-05  fsh     88-02-12  kab     88-02-13   nbh
88-01-25  emda    88-02-05  pkw     88-02-12  rap     88-02-13   rah
          .
          .
          .
```

Looking at the pacct File with acctcom

At any given time, the contents of the /var/adm/pacct? files or any file with records in the acct.h format may be examined using the acctcom program. If you don't specify any files and don't provide any standard input when you run this command, acctcom reads the pacct file. Each record read by acctcom represents information about a dead process (active processes may be examined by running the ps command). The default output of acctcom provides the following information: the name of the command (prepended with a # sign if the command was executed with super-user privileges), the user, tty name (listed as ? if unknown), starting time, ending time, real time (in seconds), CPU (in seconds), and mean size (in K). The following information can be obtained by using options: F (the fork/exec flag: 1 for fork without exec), STAT (the system exit status), HOG FACTOR, KCORE MIN, CPU FACTOR, CHARS TRNSFD, and BLOCKS READ.

The options are:

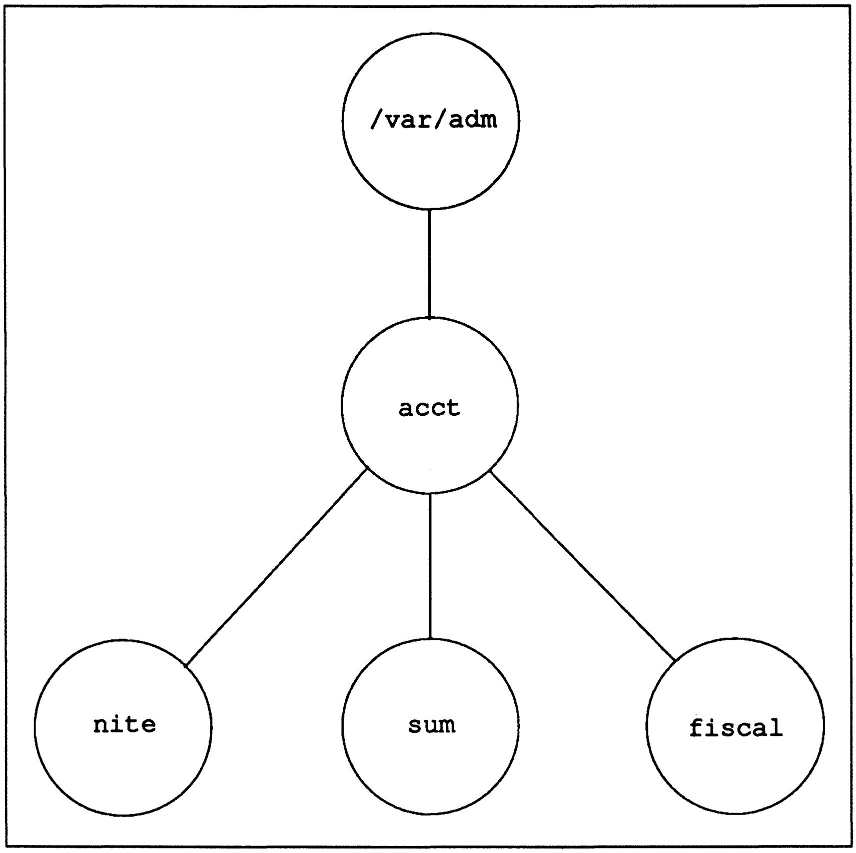
-a	Show some average statistics about the processes selected (printed after the output records).
-b	Read the file(s) backward, showing latest commands first. (Has no effect if reading standard input.)
-f	Print the fork/exec flag and system exit status columns.
-h	Instead of mean memory size, show the hog factor, which is the fraction of total available CPU time consumed by the process during its execution. Hog factor = *total_CPU_time/elapsed_time.*
-i	Print columns containing the I/O counts in the output.
-k	Show total kcore-minutes instead of memory size.
-m	Show mean core size (shown by default unless superseded by another option).
-r	Show CPU factor: *user_time/(system_time + user_time).*
-t	Show separate system and user CPU times.
-v	Exclude column headings from the output.

-l *line*	Show only processes belonging to the terminal /dev/*line*
-u *user*	Show only processes belonging to *user*.
-g *group*	Show only processes belonging to *group*.
-s *time*	Show processes existing at or after *time*, given in the format *hr*[:*min*[:*sec*]].
-e *time*	Show processes existing at or before *time*, given in the format *hr*[:*min*[:*sec*]].
-S *time*	Show processes starting at or after *time*, given in the format *hr*[:*min*[:*sec*]].
-E *time*	Show processes starting at or before *time*, given in the format *hr*[:*min*[:*sec*]]. Using the same *time* for both -S and -E shows processes that existed at the time.
-n *pattern*	Show only commands matching *pattern* (a regular expression as in ed except that "+"means one or more occurrences).
-q	Don't print output records, just print averages (akin to -a).
-o *ofile*	Instead of printing the records, copy them in acct.h format to *ofile*.
-H *factor*	Show only processes that exceed *factor*, where *factor* is the "hog factor" explained in the description of the -h option.
-O *sec*	Show only processes with CPU system time exceeding *sec* seconds.
-C *sec*	Show only processes with total CPU time, system plus user, exceeding *sec* seconds.
-I *chars*	Show only processes transferring more characters than the cut-off number specified by *chars*.

Accounting Files

The /var/adm directory structure (see Figure 2-11) contains the active data collection files and is owned by the adm login (currently user ID of 4).

Figure 2-11: Directory Structure of the adm Login



A brief description of the files found in the /var/adm directory follows:

dtmp	output from the acctdusg program
fee	output from the chargefee program, ASCII tacct records
pacct	active process accounting file
pacct?	process accounting files switched via turnacct
Spacct?.*MMDD*	process accounting files for *MMDD* during execution of runacct

The /var/adm/acct directory contains the nite, sum, and fiscal directories, which contain the actual data collection files. For example, the nite directory contains files that are reused daily by the runacct procedure. A brief summary of the files in the /var/adm/acct/nite directory follows:

active	used by runacct to record progress and print warning and error messages; active*MMDD* same as active after runacct detects an error
cms	ASCII total command summary used by prdaily
ctacct.*MMDD*	connect accounting records in tacct.h format
ctmp	Output of acctcon1 program, connect session records in ctmp.h format. (acctcon1 and acctcon2 have been replaced in UNIX System V Release 4 by acctcon; they are provided here for compatibility purposes.)
daycms	ASCII daily command summary used by prdaily
daytacct	total accounting records for one day in tacct.h format
disktacct	disk accounting records in tacct.h format, created by the dodisk procedure
fd2log	diagnostic output during execution of runacct; see "Setting Up Accounting" at the beginning of this chapter
lastdate	last day runacct executed (in date +%m%d format)
lock lock1	used to control serial use of runacct

`lineuse`	tty line usage report used by `prdaily`
`log`	diagnostic output from `acctcon`
`log`*MMDD*	same as `log` after `runacct` detects an error
`owtmp`	previous day's `wtmp` file
`reboots`	contains beginning and ending dates from `wtmp` and a listing of reboots
`statefile`	used to record current state during execution of `runacct`
`tmpwtmp`	`wtmp` file corrected by `wtmpfix`
`wtmperror`	place for `wtmpfix` error messages
`wtmperror`*MMDD*	same as `wtmperror` after `runacct` detects an error
`wtmp.`*MMDD*	`runacct`'s copy of the `wtmp` file

The `sum` directory contains the cumulative summary files updated by `runacct` and used by `monacct`. A brief summary of the files in the `/var/adm/acct/sum` directory follows:

`cms`	total command summary file for current fiscal period in internal summary format
`cmsprev`	command summary file without latest update
`daycms`	command summary file for the day's usage in internal summary format
`loginlog`	record of last date each user logged on; created by `lastlogin` and used in the `prdaily` program
`rprt`*MMDD*	saved output of `prdaily` program
`tacct`	cumulative total accounting file for current fiscal period
`tacctprev`	same as `tacct` without latest update
`tacct.`*MMDD*	total accounting file for *MMDD*

The `fiscal` directory contains periodic summary files created by `monacct`. A brief description of the files in the `/var/adm/acct/fiscal` directory follows:

`cms?`	total command summary file for fiscal period ? in internal summary format
`fiscrpt?`	report similar to `rprt?` for fiscal period ?
`tacct?`	total accounting file for fiscal period ?

Quick Reference to Accounting

- Starting accounting:

  ```
  /usr/lib/acct/startup
  ```

- Turning off accounting:

  ```
  /usr/lib/acct/shutacct
  ```

- Switching the `pacct` file to the `pacct?` file:

  ```
  /usr/lib/acct/ckpacct
  ```

- Examining the contents of `pacct`:

  ```
  acctcom
  ```

- Charging a fee:

  `/usr/lib/acct/chargefee` *login_name amount*

- Processing accounting files into a daily summary:

  ```
  /usr/lib/acct/runacct 2 > /var/adm/acct/nite/fd2log
  ```

- Doing disk accounting:

  ```
  /usr/lib/acct/dodisk
  ```

- Creating a monthly accounting report:

  ```
  /usr/lib/acct/monacct
  ```

- Printing `tacct.h` files in ASCII format:

  ```
  /usr/lib/acct/prtacct
  ```

3 Backup Service

Introduction

This chapter tells you how to perform backups so that you can always recover information that may be lost as a result of human or mechanical error. To help you plan, prepare for, and execute backups, the system provides a set of menus that guide you through the necessary steps in each process. To access the system administration menus for backups, type

```
sysadm backup_service
```

The following menu will appear on your screen:

```
1          Backup Service Management

backup     - Start Backup Jobs
history    - Backup History Information
reminder   - Schedule Backup Reminder
respond    - Respond to Backup Job Prompts
schedule   - Schedule Automatic Backups
setup      - Backup Control Table Management
status     - Backup Status Management
```

If you prefer not to use this menu, you can do the same tasks by executing shell level commands, instead. The following table shows the shell commands that correspond to the tasks listed on the menu.

Task to Be Performed	sysadm Task	Shell Command
Start backup jobs	`backup`	`backup(1M)`
Request backup history information	`history`	`bkhistory(1M)`
Schedule a backup reminder message	`reminder`	`crontab(1)`
Respond to backup job prompts	`respond`	`bkoper(1M)`
Schedule automatic backup jobs	`schedule`	`crontab(1)`
Backup table management	`setup`	`bkreg(1M)`
Backup status management	`status`	`bkstatus(1M)`

Some of the menu items in the table above have their own underlying menus.

NOTE Some of the `sysadm` forms that correspond to shell level commands may not offer the full features of that command, but will help administrators or operators with less experience by the use of prompts and help facilities.

This next table shows the menu items offered after the `history` menu item is chosen.

Task to Be Performed	sysadm Task	Shell Command
Display a full report of backup history information	`full`	`bkhistory(1M)`
Limit the size of the backup history log	`limit`	`bkhistory(1M)`
Display a selective report of backup history information	`selective`	`bkhistory(1M)`

The following table shows the menu items offered after the `reminder` menu item is chosen.

Task to Be Performed	sysadm Task	Shell Command
Add entries to the schedule of backup reminder messages	add	crontab(1)
Display the schedule of backup reminder messages	display	crontab(1)
Modify entries in the schedule of backup reminder messages	modify	crontab(1)
Remove entries from the schedule of backup reminder messages	remove	crontab(1)

The following table shows the menu items offered after the `schedule` menu item is chosen.

Task to Be Performed	sysadm Task	Shell Command
Add entries to the backup schedule	add	crontab(1)
Display the backup schedule	display	crontab(1)
Modify entries in the backup schedule	modify	crontab(1)
Remove entries from the backup schedule	remove	crontab(1)

The following table shows the menu items offered after the `setup` menu item is chosen.

Task to Be Performed	`sysadm` Task	Shell Command
Add a new backup table entry	`add`	`bkreg`(1M)
Backup exception list management	`exception_list`	`bkexcept`(1M)
Display a full backup table report	`full`	`bkreg`(1M)
Modify an existing backup table entry	`modify`	`bkreg`(1M)
Remove an existing backup table entry	`remove`	`bkreg`(1M)
Set backup table rotation period	`rotation`	`bkreg`(1M)
Display a summary backup table report	`summary`	`bkreg`(1M)

The following table shows the menu items offered after the `status` menu item is chosen.

Task to Be Performed	`sysadm` Task	Shell Command
Display a full report of pending backup requests	`full`	`bkstatus`(1M)
Limit the size of the backup status log	`limit`	`bkstatus`(1M)
Modify the status of pending backup requests	`modify`	`backup`(1M)
Display a selective report of backup requests	`selective`	`bkstatus`(1M)

Each of these tasks is explained later in this chapter. In addition, the *System Administrator's Reference Manual* provides details about each shell command and its options.

An Overview of the Backup Service

The backup service allows you to make copies of both the data on your system and the partitioning information on your disk so you can restore, automatically, all disk formats or any data later lost from your system. Subsidiary facilities provided by the backup service include tools for creating online history reports of your backups and status reports on current backup jobs.

The first part of this section, "What Is a Backup?," defines the concepts and terminology used to describe the backup service. The second part, "Backup Methods and When to Use Them," explains the various procedures available for backing up your system and gives you some guidelines for selecting the procedure most appropriate for your needs.

What Is a Backup?

A backup operation is any procedure that allows you to copy either the data on your system or the partitioning information on your disk. The data to be copied can be in the form of a file system or a data partition. The data or partitioning information to be copied is referred to as an "originating object." Your copy (or "archive volume") of an originating object is stored on media such as diskettes or cartridge tapes that are known as the "destination media." In the course of a typically large backup job, a computer operator must mount several such media on the "destination device" (that is, the tape drive or floppy drive).

Preparing for a Backup

The `backup` command requires values for a set of parameters that govern the results of a backup. These parameters are defined in a table referred to simply as the backup table. The sample backup table (`/etc/bkup/bkreg.tab`) will be useful in creating your own packup policy. For instructions on creating your own backup table, see "What Is a Backup Table?" later in this chapter.

Running a Backup

Once you have prepared the necessary backup table, you are ready to start a backup. You can run backups in either of two ways: attended or unattended. Unattended backups can be run by having `cron` run `backup` as a background command. The `backup` command, in turn, searches the backup table for all the operations to be performed at that time, and starts doing them. If operator services are needed for media handling, the operator will be contacted by UNIX system mail and will be instructed to use `bkoper`.

On the other hand, attended backups require operator interaction. If operator services are needed for media handling, the operator will be notified by system prompts (rather than by UNIX system mail). This type of backup is initiated by typing `backup -i` (described below in "Interactive Mode" under "Selecting an Operator Mode") and specifying the backup table that contains the instructions for your operation. Once you have invoked `backup`, the command will read the instructions in the specified table, along with your additional instructions.

Keeping Track of Backup Jobs

A "backup job" is a set of one or more backup operations (each of which is labeled by an identifying tag), that is begun once the `backup` command is invoked. For example, one backup job might consist of three backup operations with the tags `acct3wkly`, `serviceswkly`, and `medrecswkly`.

To help you keep track of your backups, the system keeps records of both current jobs and completed operations. Current jobs are listed in a backup status table (described later in this chapter under "Monitoring Backup Jobs"); completed jobs, in a backup history log (described in "Displaying the Backup History Log").

When a backup job is completed successfully, the job ID for it is removed from the backup status table and the operation tags associated with it are placed in the backup history log. If a backup job is not successfully completed, the job ID for it is kept in the backup status table.

Backup Methods and When to Use Them

One of the most important parameters defined in the backup table is the backup method to be used. (See "Specifying Backup Methods" under "Preparing for Backup Operations" later in this chapter for instructions on specifying a method in your backup table.) There are six methods for backing up files, directories, file systems, and data partitions: incremental file, full file, full image, full disk, full data partition, and migration backups. You can limit your backups to one of these methods or you can use several methods, at different times, to achieve a comprehensive backup strategy for your system.

Each method is characterized by how it answers the following questions:

- What type of information is copied? (For example, does this method allow you to copy files, data partitions, or both?)
- How much information is copied? (For example, does this method allow you to copy only individual files or a complete file system?)
- How is information copied? (For example, is information copied as it appears logically on the originating device or is it copied in raw format, that is, byte by byte?)
- How long does a backup take when you use this method?
- What kind of procedure must be followed to restore information that has been backed up with this method? (See the "Restore Service" chapter for details.)

You should consider each of these questions when deciding which method to use. This section describes all six methods and provides guidelines for using them.

NOTE If you are backing up the core file system, you must use either the full file backup method or the incremental file backup method; using any other method may corrupt your archive volume. For details about core file systems, see "Requesting Core File System Backups" later in this chapter.

When you have selected a backup method, request it in your backup table.

Full File Backups

This method allows you to copy all files and directories in a file system. This method is useful for backing up file systems that change frequently. It also provides more efficient restore capabilities than the full image backup and full data partition backup methods. By scheduling both full file and incremental file backups for your system, you can be sure that you have a comprehensive backup strategy.

Incremental File Backups

This method allows you to copy only those files and directories in a file system that have been modified or changed since a previous full file or full image backup. This method often requires less time and fewer destination archive volumes than other methods; the time required depends on the number of files that have been changed since the last full backup.

Full Image Backups

This method allows you to copy all the data in a file system. Unlike the full file method, the full image method allows you to copy a raw file system, byte-for-byte, onto the destination medium. This is the fastest method of doing a complete file system backup (because the files are not processed individually) but files and directories backed up in this way may take longer to restore than files and directories backed up with the full file or incremental file backup method. In addition, a spare disk partition equal in size to the origination partition is required to perform any restore operations.

Full Disk Backups

Doing backups with this method means you will be able to reinstall any required boot programs later. By using the full disk backup method, you can copy the entire contents of a disk so that if that disk is later lost, all the information required to rebuild it will be available.

The `fdisk` method backs up the disk VTOC (Volume Table of Contents) and file system characteristics, (as defined by `mkfs`(1M)). If the disk is destroyed, its contents can be recovered by first restoring the disk configuration using the `fdsk` method. Data partitions are restored using the appropriate backup method for each partition.

Full Data Partition Backups

This method allows you to copy a data partition that contains objects other than file systems, such as databases. This method is fast but it does not allow you to restore specific parts of a data partition. Because the structure of the data partition is not in the file system format, the restore service cannot identify individual files or directories; only a full data partition restore is possible.

Migrations

The migration method takes an existing archive created by another backup method and moves (migrates) it to a new location. This new location is reflected in the backup history log and any restore on the object migrated is performed using the original backup method that created the archive.

An archive created by any backup method can be migrated, as long as that archive was created on a disk partition or a file.

The migration method is useful when factors such as staffing, machine cycles, and the availability of storage devices vary over time. For example, you may want day-shift operators (who must avoid tying up too many system resources when many users are logged on) to perform quick disk-to-disk backups, and then schedule night-shift operators to migrate these backups to permanent destination devices during off-hours, when computer usage is low.

Suggestions for Performing Backup Operations

This section suggests a way to approach your backup duties. It describes the type of planning required, the steps involved in preparing for a backup, and the steps for backing up a system. Most of the effort required to use the backup and restore services is needed to prepare the backup procedures. Once this is done, performing backup and restore operations is simple.

The system administrator and computer center operator play different roles in the backup service. On small systems, one individual may perform both roles; on large systems, different people are usually assigned these areas of responsibility. The responsibilities of each are described below.

The Administrator's Tasks

The following is a detailed list of the administrator's responsibilities.

- Establish a backup plan (see "Establishing a System Backup Plan"). This plan should specify the following:
  - backup policies for a site based on factors such as resources, the needs of the users, and management directives
  - a list of file systems and data partitions that should be backed up, and for each file system or data partition, the intervals at which the backups should be done, the backup method to be used, and the types of destination devices to be used
  - how long and where destination media are to be kept before being reused
- Set up backup tables that provide the computer with instructions for implementing your plan. (See "What Is a Backup Table?" for instructions.)
- If your plan includes incremental file system backups and you want to exclude certain files from those jobs, you must create an exception list. (See "Excluding Files from Incremental Backups" under "Specifying Backup Methods," below, for instructions.)

- Either schedule backup jobs that will be invoked automatically, along with reminder messages (see "Establishing a System Backup Plan"), or invoke backup jobs manually (see the appropriate section of "Backup Methods and When to Use Them" earlier in this chapter for instructions).
- When performing backup operations, keep the following considerations in mind:
  - You must have enough destination media to complete the backup.
  - You must allow enough time to complete the backup.
- Check the status of backup jobs (see "Monitoring Backup Jobs").
- Evaluate your backup operations by examining the backup history log (see "Displaying the Backup History Log"). You may want to revise your plan on the basis of this evaluation.

The Operator's Tasks

The operator performs any attended or demand backup jobs scheduled by the administrator. During interactive backups, the operator must respond to system prompts, and mount and remove destination media.

Establishing a System Backup Plan

As an administrator, your first task regarding backups is to establish a system backup plan that specifies the following:

- Which objects need to be backed up?
- How often should the objects be backed up?
- Which is the most appropriate type of destination medium for storing the archive volume being made?
- How many destination media do you need for various backup methods (see "Previewing Backup Operations")? The number of cartridge tapes and/or diskettes you need will depend on the size of your local system. As an example of the ratio of file system blocks to backup media, suppose you need to back up the `/usr` file system on your computer and it contains 13,294 blocks. You will have to allocate either one cartridge tape or a combination of 45 diskettes for daily incremental backups and 40 diskettes for monthly full file backups.
- How much time do you need for various backup methods? As a rule of thumb, allow eight to ten times as much time in your plan for a full file backup as you do for an incremental file backup. To calculate the amount of time you will need, figure out how many destination media you will need for a particular operation (see previous item in this list), and then calculate the amount of time required when you know how long it will take to write to a certain type of medium.
- How many files are being copied during your incremental file backups? If the majority of files in a file system are being copied during incremental file backups, or if an incremental file backup takes almost as long as a full file backup, consider scheduling full file backups more frequently.
- How much time are you willing to devote to restoring a file system? If your plan relies heavily on incremental file system backups, you should keep in mind that all the destination media for each incremental backup may be needed to restore a file system to a consistent state.
- Which are the most appropriate methods for backing up an object?
- Should a backup be invoked automatically or manually?

- In what order should various backup operations be performed?
- Where and how long should backup archives be kept? You may want to save daily incremental backups for a week, weekly full backups for a month, and monthly full backups for a year or indefinitely. You may want to save some volumes off site to ensure that no one will accidentally overwrite an important archive.
- Where should the historical records of completed backups be stored?

To answer these questions, begin by observing how the resources on your computer (such as disk space and CPU time) are used. Specifically, you need to know which file systems and data partitions are used and how they are used. Consider the approximate rate of change in the file systems studied. Notice whether changes occur throughout the file system or in only a small percentage of the files. If change is frequent and widespread, it may be best to schedule nightly incremental backups and weekly full backups. If change is concentrated in just a few files, a weekly incremental backup and a monthly full backup may be sufficient.

It is also a good idea to reevaluate, periodically, your system resources and how they are used. Keep in mind that if these periodic reevaluations show that the use of your computer is changing, you may revise your backup plan.

Preparing for Backup Operations

After you have established a plan, your next job is to prepare for backup operations by setting up your backup tables. In these tables you must define the parameters that control backup jobs, such as which file system is to be backed up and the day of the week on which the backup is to be done. This section explains how to define the necessary parameters in your backup tables, how to change those parameters, and how to validate the contents of your backup tables.

What Is a Backup Table?

Each backup table defines the following parameters:

- the backup operation tag
- the rotation period (the length of time after which a backup operation should be repeated)
- the days of the week on which the backup operation is to be done
- the backup method to be used
- the priority level of the backup operation
- the origination device from which the backup is to be made
- the destination media on which the backup is to be written

The system supplies a backup table with default values that can be changed. You can use this table or you can create your own backup table with its own default values that can be changed. Either type of backup table can be set up and maintained through the `bkreg` command.

To examine the current backup table for your system (whether it is a system supplied table or a custom table), issue the following command:

```
bkreg -C fields
```

where *fields* is a list of the fields for which you want to see the existing values. Separate the items in your list with either commas or blank spaces. (If you separate items with blank spaces, enclose the entire list in double quotes.) The following are valid field names: `period`, `cweek`, `tag`, `oname`, `odevice`, `olabel`, `weeks`, `days`, `method`, `moptions`, `prio`, `depend`, `dgroup`, `ddevice`, `dchar`, and `dlabel`. If you type

```
bkreg -C tag,weeks,days,method,moptions,prio,dgroup
```

the following fields will appear on your screen (the values supplied are only examples):

```
# bkreg -C tag,weeks,days,method,moptions,prio,dgroup
Tag:Weeks:Days:Method:Options:Priority:Dgroup
::::::
::::::
::::::
::::::
rootsun:1:0:ffile:::diskette
rootdai:1:1-6:incfile:::diskette
usrdai:1:1-6:incfile:::diskette
usrsun:1:0:ffile:::diskette
```

Specifying Custom Backup Tables

The system-supplied backup table is named `/etc/bkup/bkreg.tab`. For a small system with limited backup operations, this table may be adequate. For large systems consisting of many computers, however, it may be more effective to create several backup tables of your own and sort the required backup operations into the different tables. For example, an administrator responsible for ten computers linked in a group called "services" and twelve computers linked in a group called "accounting" might set up two backup tables called `services` and `acctg`. These tables would be created by issuing the `bkreg` command followed by the `-t` option. The `-t` option would appear as follows:

```
bkreg -t /etc/bkup/services ...
bkreg -t /etc/bkup/accts ...
```

NOTE The `-t` option to the `bkreg` command can be used in combination with any other options (such as `-a` or `-e`) whenever you want to work with a custom backup table.

You can create your custom backup table in any directory, but it might be convenient to put all backup tables in the `/etc/bkup` directory, where the system-supplied backup table resides.

Assigning or Changing Default Values in a Backup Table

Whether you are using an existing backup table (either system supplied or custom made) or creating a new one, you can assign values to the fields in that table. To assign values to a backup table, run the `bkreg` command, along with the appropriate options. (See `bkreg`(1M) in the *Systems Administrator's Reference Manual*.)

NOTE If you are creating a new backup table you must identify the rotation period (by using the `-p` option) for that table before any other operations can be performed with that table. See "Specifying the Rotation Period" under "Preparing for Backup Operations" below.

Specifying Backup Methods

For each operation defined in the backup table, you must specify a backup method. This is done by issuing the `bkreg` command with the `-m` option. That option appears as follows:

`-m` *method*

where *method* is the backup method to be used. The following methods are available:

`incfile`	incremental file
`ffile`	full file system
`fimage`	full image

`fdisk`	full disk
`fdp`	full data partition

When you specify a method, you may also include any options applicable to it by using the `-b` option to `bkreg`.

The `-m` option can also be used to request a transfer of existing archives to a new location. This type of information transfer is called "migration." For more information about migrations, see "Requesting Migrations for Backed Up Information."

Method Options

When you specify a backup method (with the `-m` option to `bkreg`), you may also want to identify options that you want to have run with that method. If you then want to specify options to this method, introduce them on the command line with the `-b` flag (`-m` *method* `-b` *method options*). The following options are available:

`-d`
: Suppresses the recording of the backup in the backup history log

`-e` *filename*
: Specifies a custom exception list where *filename* is a full pathname (`incfile` and `ffile` only)

`-i`
: Excludes from the backup those files in which only the `vnode` has been changed (`incfile` only)

`-l`
: Creates a long form of the backup history log that includes a table of contents for the backup and information for each file similar to that produced by `ls -l`. (For details, see "Displaying the Backup History Log.")

`-m`
: Mounts the originating device in read-only mode before starting the backup and remounts it with its original permissions after completing the backup. This option cannot be used with the core file systems (`fimage`, `ffile`, `incfile` only).

`-o`
: Permits an operator to override label checking on destination devices (see the `-o` option to the `getvol`(1M) command and "Monitoring Backup Jobs")

`-s`	Specifies that no table of contents is to be kept on line
`-t`	Creates a table of contents for the backup on additional media instead of in the backup history log
`-v`	Validates the archive on the destination device as the backup operation is being performed to make sure that each block is readable and correct. If this check fails, the destination medium is considered unreadable. If automatic operator mode has been specified, the backup operation fails; otherwise, the operator is prompted to replace the destination medium.

Any of these options can be combined. Multiple options must be separated by blank spaces and must be enclosed in double quotes. The following is an example of how the `-m` and `-b` options appear:

```
-m ffile -b "-v -l"
```

Full File System Method

The full file system backup method copies all directories and files of a specified mounted file system to a destination device. The files are copied in the hierarchical order reflected by the directory structure.

To use this method, specify the `ffile` argument to the `-m` option on the `bkreg` command line (`-m ffile`).

Incremental File System Method

Incremental file backups copy only those files and directories that have been changed since one of the following:

- the last full file or full image backup
- the last full file or full image backup plus the last incremental file backup
- the last *n* days

Therefore, an incremental file backup must be preceded by at least one full backup that can be used as a base. Incremental file backups are useful for file systems that are changed frequently.

To use this method, specify the `incfile` argument to the `-m` option on the `bkreg` command line (`-m incfile`).

The following additional method options are available with the incremental file system backup method:

-p *mode* Shows which type of incremental file backup is to be performed, where *mode* is optional and can be any of the following:

0	the previous full file or full image
n(>0)	the last *n* days
No mode specified	The previous full file or full image

-x Ignores the exception list; backs up all changed or modified files. (For details, see "Excluding Files from Incremental Backups" below.)

Excluding Files from Incremental Backups

There are some files that are not appropriate to copy during an incremental backup. For example, it is not useful to copy temporary files, such as `/usr/tmp`, `/etc/utmp`, or `/tmp`, or files that can be reconstructed from other files, such as any `.o` files, `core` files, `a.out` files, or `nohup.out` files. In addition, files created by users, such as `dead.letter`, `junk`, `trash`, or `testing`, might not need to be backed up. Also, files that are routinely changed or created daily might not need to be backed up because they are invalid or outdated on subsequent days.

The incremental file backup method is designed to exclude these types of files automatically. Every time you run an incremental file backup, the backup command examines a list of files to be excluded. This list is known as the exception list. This list can be either the system-supplied exception list, `/etc/bkup/bkexcept.tab`, or your own exception list. (See "Creating an Exception List" below.)

You may want to review either list and make sure that it contains all the files you want to exclude from your backups and that it doesn't specify files that you want to have copied. If you want to modify the exception list, see "Modifying an Exception List" below for instructions.

If you do not want any files to be excluded from your incremental file backup (that is, if you want the exception list to be ignored), use the `-b -x` option to the `bkreg` command, when setting up your backup table.

Each entry in the exception list specifies, in the format of a "pattern," a set of one or more files. A pattern may consist of either the name of a single file or a string that includes one or more shell special characters (`\*`, `?`, or `[ ]`), and thus represents multiple files. Special characters are similar to those used in shell commands.

For details about maintaining all exception lists, see `bkexcept`(1M) in the *System Administrator's Reference Manual*.

Creating an Exception List

You can create your own custom exception list through the `bkreg` command, as shown in the following example:

```
bkreg -e acct3 -m incfile -b "-e filename"
```

where *filename* is the full pathname of the exception list to be created.

Once this entry has been created, the backup service will use this exception list for the backup selected.

Modifying an Exception List

Use the same procedures to modify either a system-supplied exception list or one you have created. The `-t` option to the `bkexcept` command must be used each time an exception list other than the default list is referenced. If the `-t` option is not used, the default list is referenced.

To add entries to an exception list, type

```
bkexcept -a pattern
```

where *pattern* is a string that represents a file or a set of files, as defined on the `bkexcept`(1M) page in the *System Administrator's Reference Manual*. Items in a list of patterns must be separated by commas or by blank spaces (items separated by spaces must be enclosed in double quotes). For example, suppose you want to add entries for the following items (that is, you want to exclude these items from your incremental backups):

- all subdirectories and files under the `/tmp` directory
- all subdirectories and files under the `/usr/tmp` directory
- any file named `junk`
- the user file named `/usr/accts/clerk3/oldfile`

Type the following command line:

```
bkexcept -a \
/tmp/\*,/usr/tmp/\*,\*/junk,/usr/accts/clerk3/oldfile
```

NOTE If a command does not fit on one line, escape the newline character between multiple lines by entering a backslash (\) at the end of every line except the last. If the command syntax requires a space between elements in the command, leave a space before each backslash.

As shown in this example, special characters (such as *) must be preceded by a shell escape character (such as the backslash shown above). The escape character prevents the shell from expanding the special character. If you prefer not to use escape characters in your command line, enclose the string of arguments to -a in quotation marks, as follows:

```
bkexcept -a \
``/tmp/\*,/usr/tmp/\*,\*/junk,/usr/accts/clerk3/oldfile"
```

Another way to avoid using escape characters on the command line is by entering a list of patterns from standard input. To do this, first enter a dash in place of the pattern argument on the command line. Then specify the desired patterns on separate lines. To end your list, type CTRL-d. The following example shows how to specify patterns in this way.

```
$ bkexcept -t /etc/save.d/except -a -
/tmp/*
/usr/tmp/*
/usr/spool/*
*/trash
/usr/accts/clerk3/oldfile
CTRL-d
$
```

To remove an entry from your exception list, type

```
bkexcept -r pattern . . .
```

where *pattern* matches an entry in the exception list. For example, to remove `/usr/spool/*` and `/usr/rje/*` from the exception list, type the following:

```
bkexcept -r /usr/spool/\*,/usr/rje/\*
```

You can remove entries by listing *patterns* on a command line, as shown above, or you can remove entries by specifying them on separate lines, by using a dash for the value of *pattern*. To end your list, type CTRL-d, as shown in the following example:

```
$ bkexcept -r -
/tmp/*
/usr/tmp/*
/usr/spool/*
*/trash
/usr/accts/clerk3/oldfile
CTRL-d
$
```

When you have added or removed entries in the exception list, you may want to see what the list contains. To display the full contents of the exception list (in ASCII collating order), invoke `bkexcept` with no options. To tailor the display, type

```
bkexcept -d patterns
```

The following example shows how a display appears on your screen. If you type

```
bkexcept -d /usr,/usr/spool
```

your output might appear as follows:

```
Files in the exception list beginning with /usr:
/usr/at
/usr/games
/usr/jpv/testfiles
/usr/spool/crontab

Files in the exception list beginning with /usr/spool:
/usr/spool/crontab
```

For exception list files containing only a RETURN character, `bkexcept -d /usr` returns successfully with a null exception list. For files of zero length (no characters), `bkexcept -d /usr` returns the message

```
search of table failed
```

Converting Exception Lists from Earlier Backup Services

Prior versions of the backup service created exception lists using `ed` syntax. The `bkexcept -C` command translates the entries in these earlier exception lists to the new shell pattern format. The translation is not perfect; not all `ed` patterns have equivalents in shell patterns. For those patterns that have no equivalents, an attempt at translation is made, and the translated version is flagged with the word `QUESTIONABLE`.

Because the translation is not perfect, you need to edit the output of the `bkexcept -C` command before adding it to the default exception list for the current backup service. The following procedure explains how to do this.

Step 1 The translation of the exception list is directed to standard output. Redirect the standard output to a file, such as `checkfile` in the following example:

```
bkexcept -C /etc/save.d/except > checkfile
```

The exception list from the prior version of the backup service specified in this example is `/etc/save.d/except`. Before being converted, the contents of this file appear as follows:

```
# Patterns of filenames to be excluded from saving by savefiles.
# These are ed(1) regular expressions.
/.news_time$
/.yesterday$
/a.out$
/core$
/dead.[a-1]*$
/ed.hup$
/nohup.out$
/tmp/
 .o$
^/etc/mnttab$
^/etc/save.d/timestamp/
^/etc/utmp$
^/etc/wtmp$
^/usr/adm/
^/usr/at/
^/usr/crash/
^/usr/dict/
^/usr/spool/
^/usr/tmp/
```

After this file has been converted by `bkexcept -C`, and you have redirected the output to `checkfile`, the contents of `checkfile` appear as follows:

```
QUESTIONABLE: #  Patterns of filenames to be excluded from saving by savefiles.
QUESTIONABLE: #  These are ed(1) regular expressions.
*/.news_time
*/.yesterday
*/a.out
core
*/dead.[a-l]*
*/ed.hup
*/nohup.out
*/tmp/*
*.o
/etc/mnttab
/etc/save.d/timestamp/
/etc/utmp
/etc/wtmp
/usr/adm/*
/usr/at/*
/usr/crash/*
/usr/dict/*
/usr/spool/*
/usr/tmp/*
```

Step 2 Review the contents of the new file (`checkfile` in this example). Notice that the `QUESTIONABLE` flag is used for two purposes: to mark comments in the old file, and to draw your attention to entries that may not have been translated properly. Delete the `QUESTIONABLE` flags that precede comments, preserving any comments that you want.

Review the entries that may not have been translated properly. Revise those entries that are not in the correct format by deleting the `QUESTIONABLE` flag and modifying the pattern as necessary. If you decide that a translation is adequate, you need only remove the `QUESTIONABLE` flag.

Step 3 After editing the entries in the converted file (`checkfile` in this example), add the contents of this file to the current exception list (`/etc/bkup/bkexcept.tab`). To do this, specify the converted file on the `bkexcept -a` command line, as shown in the following example:

```
bkexcept -a - < checkfile
```

Full Image Method

A full image backup copies an entire file system, byte-for-byte, starting with the first block and ending with the last. A full image backup differs from a full file backup because it does not copy the file system according to its directory structure. Instead, a full image backup copies the data blocks in the order in which they appear on the disk, from the first segment to the last segment.

To use this method, specify the `fimage` argument to the `-m` option on the `bkreg` command line (`-m fimage`).

Full Disk Method

The full disk backup method allows you to copy all the information required (by the restore service) to recover the format of an entire disk. Typically, the disk format is restored, followed by individual file systems and, finally, by the data partitioning information.

To use this method, specify the `fdisk` argument to the `-m` option on the `bkreg` command line (`-m fdisk`).

Full Data Partition Method

A full data partition backup allows you to copy a data partition that contains objects other than file systems, such as databases. Also, it allows you to copy a raw data partition, byte-for-byte, starting with the first block of the data partition and ending with the last.

To use this method, specify the `fdp` argument to the `-m` option on the `bkreg` command line (`-m fdp`).

Requesting Migrations for Backed Up Information

Migration is the process of moving an existing archive to a new location. The archive may have been created by any backup method and its new location is recorded in the backup history log. Any restore operation done with a migrated object must be done with the restore method appropriate for the backup method by which the archive was originally created.

A migration cannot be done until a backup operation has been performed. The fact that an operation has been performed implies that an entry for that operation exists in the backup table. That entry includes values for the originating device and the destination device.

Migrations are useful when factors such as staffing, machine cycles, and the availability of destination devices vary over time.

For example, you may want to schedule an automatic incremental file system backup to a spare disk partition at a specified time, and later have an operator move the resulting archive to tape. Specifically, suppose you want an incremental file backup for the operation known as `mktg3`, specifying `/usr:/dev/dsk/c1d0s2:usr` as the originating device and `:/dev/dsk/c1d1s2:` as the destination device. To request this operation, you must add an entry to the backup table. Issue the `bkreg` command with the options shown below:

```
bkreg -a mktg3 -m incfile \
-o /usr:/dev/dsk/c1d0s2:usr \
-d :/dev/dsk/c1d1s2:
```

Once this entry exists in the backup table, and `crontab` is updated to run `backup` at the required time, the information saved by the backup (the archive) will be stored in the designated destination device (`:/dev/dsk/c1d1s2:`).

To migrate the archive you must edit the entry for the relevant operation in the backup table, specifying a migration and the new destination for that archive. Therefore, you must edit the entry for the `mktg3` operation by issuing a command such as the following:

```
bkreg -e mktg3 -m migration \
-o :/dev/dsk/c1d1s2: \
-d diskette::cap=1422:mktgwklyA,mktgwklyB
```

As shown here, the destination device specified for the incremental backup is also specified as the originating device for the migration.

Once you have edited the specified entry in the backup table, an operator can request the migration with the `backup` command, as long as the migration requested is limited to the originating device you have specified in the table. In this example scenario, the operator now enters the following command:

```
backup -i -o :/dev/dsk/c1d1s2:
```

The results of a migration are as follows:

- An existing archive is stored on a new destination device (from which it can be restored in the same way any other archive is restored).
- The backup history log is updated to show the new location of information that has been migrated.
- The table of contents (for the archive that has been migrated) is updated to show the new location of the archive.

Requesting Core File System Backups

Core file systems are different from other file systems because they contain system software and, therefore, they must always remain mounted even when they are being backed up and restored. (For a description and list of core file systems, see the "File System Administration" chapter.)

A backup done on a core file system should be done on a "demand only" basis; specify `demand` in the `bkreg` table entry for each core file system backup.

Unlike other file systems, core file systems must be backed up only with the `-m ffile` option or the `-m incfile` option to the `bkreg` command. Do not use the full image backup method on a core file system because changes made to files in this system during this type of backup can corrupt the resulting archive. Running an incremental file backup or a full file backup is less dangerous to a core file system because changes made to a core file system during one of these types of backups will not corrupt your destination archive; at worst, you may get an intermediate file copy, or one that is not up to date.

Suppose you want to add an entry with the tag `usrdai` to the default backup table. The originating object to be backed up is the `/usr` file system on the `/dev/rdsk/c1d0s2` device (which is labeled `/usr`). You want to use the incremental file backup method (with the `-m incfile` option) on the next available diskette device using the three diskette volumes `usrdai1`, `usrdai2`, and `usrdai3`. (These volumes have a capacity of 1422 blocks.) Type the following command line:

```
bkreg -a usrdai -o /usr:/dev/rdsk/c1d0s2:usr \
    -c demand -m incfile \
    -d diskette::cap=1422:usrdai1,usrdai2,usrdai3
```

See "Incremental File Backups" under "Backup Methods and When to Use Them" for information on the options listed in the above example.

Specifying Originating Objects

You can define the originating object for a backup operation by using the `-o` *orig* option to the `bkreg` command.

The *orig* argument takes the following form:

oname:*odevice*[:*omname*]

The following is an example of the `-o` *orig* option:

```
-o /usr:/usr/dev/dsk/c1d0s0:usr
```

Each component of the argument *orig* is defined below:

oname (`/usr`)	The pathname of the originating object. For file system partitions, this is usually the node name on which the file system is mounted. For data partitions, it is any valid pathname. This value is provided to the backup method and validated by the `backup` command. The data partition backup methods (invoked with the `-m fdp` and `-m fdisk` options to `bkreg`) do not require the name of the originating object; the `ffile` and `incfile` methods require *oname*.

odevice	The raw disk partition device name for the originating object.
omname	The volume label for the originating object. For file system partitions, it corresponds to the *volumename* specified with the `labelit` command. A data partition may have an associated volume name. If it does, the name is known only externally (taped on the device); run the `getvol`(1M) command to validate the name. Not all file system types support *omname*. See the "Storage Device Management" chapter for a list of those that do. (For details about volume names and the `labelit` command, see the "Storage Device Management" chapter.)

Specifying Destination Devices

All backup operations require a destination device on which an archive volume can be stored. To specify a destination device, use the `-d` option, as follows:

`-d` *ddev*

where *ddev* takes the form

dgroup:ddevice[*:dchar*][*:dmnames*]

Here both *dgroup* and *ddevice* must be specified and *dchar* and *dmnames* are optional. Colons separate fields and must be included as shown above. *dgroup* is the device group for the destination device. (For a description of device groups, see the "Storage Device Management" chapter.) If *dgroup* is not specified, *ddevice* must be specified and any available destination device in *ddevice* will be used.

dchar describes the characteristics of a destination device and, if specified, it overrides the default characteristics for the device and group. These characteristics are found in `/etc/device.tab`. The critical characteristics are type and capacity; these should be specified with *dchar*. (For details about the format and meaning of *dchar*, see the "Storage Device Management" chapter.)

NOTE

If device characteristics for a backup or restore device are not specified in `/etc/device.tab`, they must be specified in the backup register table.

dmnames is a list of names of the destination media. The items in this list must be separated either by commas or by blank spaces (items separated by blanks must be enclosed in double quotes). Each name in the list corresponds to a *volumename* specified with the `labelit` command. (For details about destination device labels and the `labelit` command, see the "Storage Device Management" chapter.) If *dmnames* is omitted, the `backup` and `restore` commands do not validate the labels on the destination devices.

Specifying the Rotation Period

A rotation period is the number of weeks between invocations of a backup operation. The rotation period for every backup operation is assumed to be one week (the default period) unless it is specified otherwise in the backup table. To set the rotation period for the system-supplied backup table, run

-p *rperiod*

where *rperiod* is an integer between 1 and 52 that represents the number of weeks between backups. (To set the rotation period in a custom backup table, use the -t option with the -p option.)

NOTE

The -p option to `bkreg` cannot be used with any other options on the same command line.

By default, a rotation period always begins on a Sunday, but you can change that parameter by entering

-c *weeks:days*

where *weeks* is a list of one or more integers between 1 and 52, that cannot exceed the value set by the -p *period* option. *days* is a list of either integers between 0 (Sunday) and 6 (Saturday) or the characters `s, m, t, w, th, f,` and `sa`. Both the *weeks* argument and the *days* argument can be specified as either a list of individual items (such as `1, 3, 5`) or as a range of items (such as 1-3). The items in each list can be separated by either commas or blank spaces (in which case the list is enclosed in double quotes).

For example, if you want a backup operation to be performed every Sunday, Tuesday, Wednesday, Thursday, and Saturday on the first, second, third, and fifth weeks of the year, specify these times as shown below:

```
bkreg -a svcs -m incfile -c 1-3,5:s,t-th,sa
```

Another way of defining the rotation period for a backup operation is by requesting that this operation be performed only on a demand basis. Operations for which this request has been made are not performed regularly; they can be performed only when the `backup` command is invoked with `-c demand`. The following example command line shows how to request demand only status for all the backup operations listed in the `services` table.

```
bkreg -c demand -t /etc/bkup/services
```

Establishing Dependencies and Priorities

Some backup operations should be started before others begin. Some backup operations should not be run at all until other backups have been completed. The backup service lets you handle both these situations by providing a way for you to establish priorities and dependencies when setting up backup tables.

You may want to list your backup operations in the order you want them to run. To do this, assign a priority level to each backup operation defined in the backup table by using the `-P` option and specifying a priority level. The priority level is an integer from 0 to 100 where 0 is the lowest priority and 100 is the highest priority. If a set of backup operations are to be performed at the same time, each backup operation is not started until all others with a higher priority are completed. All backup operations with the same priority can be done simultaneously, unless the priority is 0. All backups with a priority of 0 are performed sequentially in an unspecified order.

You can also specify that a backup operation not be started until a set of other backup operations is completed successfully. These dependencies must be identified in the backup table. To add a list of dependencies to the backup table, run the `-D` option, as follows:

```
-D depends
```

depends is a list of the operation tags for the operations on which a particular backup operation depends. Items in the list must be either separated with commas or separated with blank spaces (items separated by blanks must be enclosed in double quotes).

Establishing dependencies is particularly useful when using the migration method. A backup operation's dependencies take precedence over a backup operation's priorities. For example, an administrator may have a backup operation called `acctswkly` that is dependent on completion of the backup operation `SysengFri`. However, `SysengFri` has a priority of 40, which is less than the priority of `acctswkly1` (50). According to the rules of priorities, `SysengFri` should not begin before `acctswkly1`. But because dependencies take precedence over priorities, the `SysengFri` backup operation will be performed before the `acctswkly1` backup. To check the priorities and dependencies of these backup operations, type

```
bkreg -C tag,priority,depends
```

The following is an example of how the information would appear:

```
acctswkly1:50:SysengFri
SysengFri:40:
```

Creating Tables of Contents

Another item that you can specify in the backup table is a table of contents that lists the files and directories on a particular destination device.

NOTE A table of contents is used by the restore service to locate files and directories to be restored.

Tables of contents can be provided only for backup operations performed with the incremental file, full file, or full image backup methods. Tables of contents are created and changed by using the `-s` and `-t` arguments to the `-m` *method* option. Because arguments to the `-m` *method* option must always be introduced on the command line by the `-b` flag, you would enter the `-s` and `-t` arguments as follows:

```
-m ffile -b "-s -t"
```

You can specify either a long-form or a short-form table of contents. The short form of a table of contents shows the names of the directories or files, and volumes that have been stored on a particular destination device. The long form of the table contains the same information as the short form, plus the information about those directories or files that is normally provided by the `ls -l` command.

By default, the system creates the short form of the table of contents. If you require the long form, specify the `-l` argument after the `-b` flag, as follows:

`-m` *method* `-b -l`

method may be `ffile, incfile,` or `fimage.`

You can store a table of contents in any of three ways:

- online
- on removable destination volumes
- both online and on destination volumes

Alternatively, you can specify that no table of contents be stored. The location of a table of contents is controlled by three factors: the system default, the `-s` argument to `bkreg -b`, and the `-t` argument to `bkreg -b`. These three factors can be used in any of the following combinations:

- To store a table of contents online only, use the system default; do not specify `-s` or `-t` after the `bkreg -b` flag.
- To store a table of contents on removable destination devices only, specify both `-s` and `-t` after the `-b` flag, as follows:

  `-m` *method* `-b "-s -t"`

- To store a table of contents both online and on removable destination devices, specify `-t` after the `-b` flag, as follows:

  `-m` *method* `-b -t`

- To request that no table of contents be stored, specify `-s` after the `-b` flag, as follows:

  `-m` *method* `-b -s`

Adding or Changing Backup Table Entries

There are three ways to change the contents of a backup table: you can add a new backup operation to the table, modify the instructions for a backup operation already defined in the table, or remove a backup operation from the table.

Adding an Operation Entry

To add a new backup operation to the table, type

```
bkreg -a tag
```

where *tag* identifies a backup operation. The `-a` option must be followed by the `-o, -c, -m,` and `-d` options and, if you choose, any of the following options that are not required: `-b, -t, -P,` and `-D`. The following table summarizes the options to the `-a` option.

Option and Argument	Argument Form	Meaning
`-a` *tag*	Alphanumeric string of any length	Adds a new entry to the backup table. The `-a` option must be followed by the `-o, -c, -m,` and `-d` options. You can also use the `-b, -t, -P,` and `-D` options; if you do not, the default values for those options are used.

The following example shows how to add an entry to a custom backup table.

```
bkreg -a acct5 -o /usr:/dev/rdsk/c1d0s2:usr \
-c 1,3-10,13:th -m incfile -b "-t -x" \
-d diskette::cap=1422:acctwkly1,acctwkly2,acctwkly3 \
-t /etc/bkup/wklybu.tab
```

This command line allows you to add an entry for a backup operation named acct5 to a backup table named wklybu.tab. (If wklybu.tab does not already exist, it will be created.) The originating object to be backed up is the /usr file system on the /dev/rdsk/c1d0s2 device. The backup operation defined here will be performed every two weeks on Sunday using the incremental file backup method.

The method options specify that a table of contents will be created on a destination device. The backup will be done to the next available destination device using the devices labeled acctwkly1, acctwkly2, and acctwkly3. These devices have a capacity of 1422 blocks each.

Modifying an Existing Operation Entry

To modify the contents of a backup operation already defined in a backup table, use the -e option, followed by a tag which identifies the existing backup operation you want to modify and any other options for which you want to change the value. The following example command line shows how to do this:

```
bkreg -e acct5 -t /etc/bkup/wklybu.tab \
-o /usr:/dev/rdsk/c1d0s2:usr -m incfile -b "-t -x" \
-d diskette::cap=1422:acctwkly1,acctwkly2,acctwkly3
```

Option and Argument	Argument Form	Meaning
-e *tag*	Alphanumeric string	Allows you to edit an existing table entry. If any of the options -b, -c, -m, -o, -D, or -P are present, the arguments to them replace the current values for the specified entries in the table.

Removing an Operation Entry

To remove the entry for an operation from a backup table, type

```
bkreg -r tag
```

where *tag* specifies the tag of the backup operation you want to remove.

Option and Argument	Argument Form	Meaning
`-r` *tag*	Alphanumeric string	Removes the specified entry

Validating Backup Tables

Before the operations listed in a backup table can be performed, the backup service checks for the consistency of several items (such as partition information) between the destination device and the backup table. Consistency is validated when the `backup` command is invoked. If `backup` is invoked and any of the consistency checks fail, `backup` terminates. If you prefer to validate the consistency of items in the system-supplied backup table manually, issue the following command:

```
backup -n -e
```

This command processes the backup operation without actually running it. By doing this step before running the backup, you can avoid a `backup` failure. (You can perform the same step for a custom backup table by entering the `-t` *table* option after the above command.)

If you want to check your backup tables before requesting a validation check, you can request a display of the contents of your tables. A display consists of a set of entries, each of which defines a backup operation. The following fields are available for display:

rperiod	the number of weeks in the rotation period
cweek	the week in the rotation period to which the current week corresponds

tag	a unique identifier associated with the backup operation
oname	the pathname of the originating object
odevice	the device name of the originating device
olabel	the volume label of the originating device
week	the weeks, during the rotation period, in which the backup operation is performed
day	the days of the week on which the backup is performed
method	the backup method used
moptions	the options associated with a particular backup method
priority	the priority level for this backup operation
depends	tags for backup operations that must be completed successfully before this backup operation can begin
dgroup	the device group for a destination device
ddevice	the device name of a destination device
dchar	characteristics of a destination device
dmname	the volume labels for a destination device

To display a complete table, run the `bkreg` command with the `-A` option. The `-A` option can be followed by the `-h`, `-s`, `-v`, `-t`, or `-c` options.

The output for this command is a set of extremely long lines; it is best used as input to a filter. To obtain a display that is easier to look at, run the following options to `bkreg`:

```
bkreg -C fields
```

Specify only those fields from the list above that you want to see. For example, you may want to display only the backup tags, the weeks of the rotation period, the backup methods used, dependencies, and priorities. If you enter

```
bkreg -C tag,week,method,depend,prio
```

the following information is displayed:

```
Tag:Weeks:Method:Depends:Pri
rootsun:1-8:ffile::
rootsp:r,8:ffile::20
usrdai:1-8:ffile::10
usrsun:1-8:ffile::
medrecs3:demand:ffile:mainofc:4
mainofc:demand:ffile::6
newusr2:demand:ffile::2
```

The `-C` option can be followed by one or more of the following options: `-h`, `-v`, `-t`, `-F`, or `-c`.

You may also tailor the information displayed by using either the `-O` or the `-R` option to the `bkreg` command. The `-O` option allows you to display a summary of all originating objects in the table. The `-R` option accesses a summary of all destination devices in the table. The `-O` and `-R` options, like the `-A` option, can be followed by one or more of the following options: `-h`, `-s`, `-v`, `-t`, and `-c`.

Performing Backup Operations

Once you have finished setting up your backup tables, you are almost ready to start running backup operations. Before you do, you will need to answer three questions:

- Which operator mode do you want to use?
- How much destination device space or how many destination archive volumes will you need?
- Do you want to limit your backup operation to a subset of the information normally copied during a defined rotation period (such as only today's files)?

This section defines each of these questions and explains how to find answers to them.

After you have selected an operator mode, have set aside the required number of destination volumes, and have decided which, if any, of the backup table values you want to override, you are ready to begin. You have already requested a backup method in your backup table. The rest of this section provides detailed descriptions of running backup operations using each backup method. It also explains how to back up a core file system (core file systems have special needs not associated with other file systems).

Selecting an Operator Mode

Once your backup tables are created, you must decide whether your backup operation requires operator assistance. Usually an operator is needed to mount destination devices. Some backup operations, however, are small enough that they can be done without any help from an operator. To accommodate both situations, the backup service allows you to run backup operations attended or unattended. For an attended backup, an administrator (or operator) issues the backup command and provides any necessary assistance during the course of the backup job.

Unattended backups are useful if your backup jobs do not require an operator to mount multiple destination devices and if you want your site's backup jobs to be done during off-hours when the computer center is unstaffed.

An unattended backup operation is run without the help of an operator; it is invoked by the system at a time you have specified in the backup table. Attended backups are useful when operators are present and when you want flexibility in the time of day that backup jobs occur.

There are three operator modes to accommodate situations in which:

- an operator is available but not at the terminal (background backups — the default mode)
- an operator is available at the terminal throughout a backup (interactive backups)
- no operator is available (automatic backups)

This section describes each mode and explains when you might want to use it.

Background Mode

If backup is invoked with no options, the background mode is used by default (this mode is normally set up to be run by cron). In the background mode, whenever a backup operation requires an operator's assistance, it sends a mail message to the operator's mailbox. The mail message reads as follows:

```
The following backup requires operator intervention:

job_id     tag      time     volume
back-441   acct3    09:35    /usr:/dev/rdsk/c1d0s2:usr
```

When a new medium is needed for a backup operation running in background mode, the backup operation is suspended until the operator receives the mail, issues the bkoper command, and responds to the prompts, thereby enabling the job to proceed. Note that in this mode, this type of suspension delays completion of any scheduled backup operations that depend on this backup or that have a lower priority.

Interactive Mode

If an operator will be present at the terminal during backup jobs, you may want to run your jobs in interactive mode. When you use this mode, the system sends all prompts directly to the standard output of the terminal where the `backup` command was issued. Interactive mode allows the operator on duty to respond to the prompts as they arrive at the operator terminal, to insert or remove destination media as required, and to oversee the progress of the backup operation. To request interactive mode, type

```
backup -i
```

If an operator will be present at the terminal during backup jobs, and wants to monitor an incremental file backup or a full file backup closely, the operator can request the backup service to post the progress of the operation. This request is made by running a backup job in verbose mode. This mode is a form of interactive mode, so to request it, use both the `-i` and the `-v` options, as follows:

```
backup -iv
```

When a backup is run with this command line, the name of every file and directory being backed up is displayed on standard output.

If you want to track the progress of a backup in more detail, request special verbose mode with the `-s` option:

```
backup -is
```

When a backup is run with this command, a dot is displayed as every 100 (512-byte) blocks are transferred to the destination device.

Automatic Mode

If no operator is available either to respond at the terminal or to receive mail messages, you can run your backup job in automatic mode. This mode allows you to schedule jobs in advance and to arrange for the system to start them without operator assistance at scheduled times. If any single backup operation requires operator assistance, that operation fails and the rest of the scheduled backup operations continue. To request automatic mode for your backup job, type

```
backup -a
```

Previewing Backup Operations

There may be times when you want to know the schedule of backup operations for a day without invoking any jobs. The backup service provides two preview capabilities that allow you to (*a*) preview the set of backup operations for a day, or (*b*) get an estimate of the number of destination media required for a backup.

To display the current day's backup operations in the order that they would proceed if invoked (that is, according to priorities and dependencies), type

```
        backup -n
```

The following is an example of how information may appear:

```
  Tag     Orig.Name   Orig.Device     Dest.Group   Dest.Device    Pri  Depends On
 usrdai     /usr    /dev/dsk/c1d0s2   diskette   /dev/diskette   10
 usr2dai    /usr    /dev/dsk/c1d0s8   diskette   /dev/diskette   10     usrdai
 rootdai     /      /dev/dsk/c1d0s0   diskette   /dev/diskette    0
```

You can preview the same information for any day in a rotation period by adding the `-c` *week:day* option, as shown in the following example:

```
        backup -n -c 3:f
```

This command line requests a list of backup operations scheduled for Friday (`f`) of the third (`3`) week in the rotation period.

After previewing the day's schedule of backups, you should find out whether you have enough space on your destination device or archive volumes before initiating a backup. You can find out how may devices you need by using the -ne option to `backup`.

The -ne and -c options can be combined to obtain a report that lists the scheduled backup operations for a specified day, along with an estimate of the number of devices required for each operation.

In addition to providing this information, the -ne option validates the consistency between corresponding items in the backup table and on the destination device. If any of the validation checks fail, the service sends an error message to standard error. This capability enables you to correct problems before initiating an actual backup.

Requesting Limited Backups

When you issue the `backup` command, usually all operations defined in your backup table for the current rotation period are performed. There may be times, however, when you want to run a backup without having all operations performed. For example, you may want to run only backup operations defined for `demand` (that is, operations that are not scheduled to be run regularly). This section describes ways in which you can limit the backup operations to be performed.

- To run only those backup operations scheduled for the current day, use the `backup` command without the `-c` option. To invoke backup operations scheduled for a day other than the current day, type

  backup -c *week:day*

  specifying values (integers) for the desired week of the rotation period and values (either integers or letters) for the day. For example, if the current week is the eighth week of the rotation period, but you want to run the backup operations scheduled for Thursday of the seventh week, invoke:

  ```
  backup -c 7:th
  ```

- To run backups that are scheduled to run only on demand, invoke

  ```
  backup -c demand
  ```

- To run backup operations defined in a custom backup table, type

  backup -t *table*

- To back up objects on a particular originating device, type

  backup -o *orig*

  where *orig* must be of the form *oname*:*odevice*: [*omname*].

- To request that mail be sent to a specified user when the entire backup operation is complete, invoke

  ```
  backup -m user
  ```

The above options can be used in various combinations on the `backup` command line. For example, suppose you want to invoke those backups listed on your custom backup table (`/etc/bkup/accts.tab`) that are scheduled for Friday of the second week in the rotation period. In addition, when the backup job has been completed, you want mail to be sent to the user with login `supv3`. Invoke this operation by typing

```
backup -t /etc/bkup/accts.tab -c 2:f -m supv3
```

Monitoring Backup Jobs

Backup jobs may require operator assistance for such tasks as mounting diskettes, cartridge tapes, and 9-track tapes and checking the labels on destination media to verify that the correct volumes are being used. An operator may perform a backup in any of three modes: background mode, interactive mode, or automatic mode. (See "Selecting an Operator Mode" for descriptions of the three modes.) This section explains how an operator interacts with a backup operation being run in background mode.

When a backup job in background mode cannot proceed further without the assistance of an operator, the `backup` command sends a `mail` message to the operator requesting assistance for that job. To find out what needs to be done, the operator must type `bkoper`. The `bkoper` command responds by printing a list of backup jobs for which assistance is needed, such as those shown in the following example:

```
1. back-111 usrsun /dev/dsk/c1d0s1  disk /dev/dsk/c2d1s9 usrsave
2. back-112 fs2daily /dev/dsk/c1d0s8 ctape /dev/ctape1-
```

Each entry contains the following: operation number (the initial digit followed by a period), backup job ID (the `back-`*nnn* label), operation tag, originating device, destination device group, destination device name, and archive volume label. In the example above, the dash displayed as the last item of the second entry shows that no specific volume label is required for the backup operation listed. Backup operations are numbered in the order in which they appear in the backup table.

The `backup` command then displays the following question:

```
Which prompt do you want to respond to?
Type [q] to quit bkoper
Type [h] to display the list of backup operations
Type a number or RETURN to service a backup operation
?
```

To find out what kind of assistance is needed for the first operation listed, press **RETURN** or type `1`.

The bkoper command will respond by explaining the task that needs to be done. For example, if you press RETURN after seeing the list displayed above, you might receive a message such as the following:

```
Insert a diskette into the floppy drive.  The diskette should
be internally labeled as follows:

        usrsave

Type [go] when ready
  or [f] to format the diskette
  or [q] to quit.
```

The operator can mount the volume shown (usrsave) or mount an unlabeled volume. In addition, if the -b -o method option has been used with this backup, the operator can override the request and mount any volume, regardless of whether or not it has been used before. If the -b -o method option has been used, the previous display will appear as follows:

```
Insert a diskette into the floppy drive.  The diskette should
be internally labeled as follows:

        usrsave

Type [go] when ready
  or [f] to format the diskette
  or [o] to use the current label anyway
  or [q] to quit.
```

After performing the task requested, press RETURN to find out what kind of assistance is needed for the next operation listed.

If you want to service an operation other than the current one (that is, other than the one at the top of the list), you can do so with either the p (print) keyletter or the t (type) keyletter, followed by the number of the relevant operation. For example, if you want to service the second operation before the current one, type

```
p2
```

These are only a few of the responses you may enter after the prompt, `Which prompt do you want to respond to?` For a complete list of possible responses to this question, see `bkoper`(1M) in the *System Administrator's Reference Manual*.

If, after the original list of operations has appeared, servicing is required for other operations (and you are still interacting through the `bkoper` command), the following message appears:

```
There are new backup operations requiring service.
```

When you have finished servicing all the operations that require assistance, the following message is displayed:

```
No more backup operations are waiting for operator action at
  this time.
```

If you want to interrupt your session with `bkoper` to perform a task at the shell level, type `!` and enter the desired command. To quit the `bkoper` session altogether, type `q`.

Checking Job Status

You can check the status of backup jobs (and the backup operations included in each job) by using the `bkstatus` command.

Each backup job progresses through the six states listed below.

pending	The `backup` command has been invoked and the operations listed in the backup table for the specified day are scheduled to occur.
active	All backup operations have been assigned destination devices and archiving is currently underway, or a suspended backup has been resumed.

waiting	The backup job is waiting for operator assistance, such as the mounting of a new tape.
suspended	The backup job has been suspended by an invocation of `backup -S`.
failed	The backup job has failed or has been canceled.
completed	The backup job has been completed successfully.

The status of backup operations is recorded in the `/etc/bkup/bkstatus.tab` table. In the table, each state is represented by the first letter of the relevant status: `p` (pending), `a` (active), `w` (waiting), `s` (suspended), `f` (failed), or `c` (completed).

You may display the backup status table in any of several ways. For all information about backup operations that are in progress (those labeled `a`, `p`, `w`, or `s`), invoke `bkstatus` with no options. To include information about backups with a status of `f` (failed) or `c` (complete), enter

```
bkstatus -a
```

A display such as the following will appear on your screen:

```
  Jobid      Tag     User      Oname       Odevice      Start Time      Dest    Status
--------------------------------------------------------------------------------------
back-459   UsDly    oper1  /usr/spool  /dev/c1d0s8        -         diskette     f
back-459   PtsDly   oper1   VTOCc1d0   /dev/c1d0s7        -         diskette     c
back-395   SysDai   oper2     /sys     /dev/c1d0s9  May 26 16:45   diskette     c
```

If a full report is not required, you can limit the types of information that are displayed. For example, you can restrict a report to information about only specified jobs by invoking

`bkstatus -j` *jobids*

This command will not show those operations that were completed or failed. In addition, all *jobids* must be of the form `back-`*number*.

To restrict a report to information about jobs in particular states, invoke

```
        bkstatus -s states
```

where *states* is a list of key-letters that are concatenated, comma-separated, or blank-separated (items separated by blanks are enclosed in double quotes), as shown in the examples below:

- `apf`
- `a,p,f`
- `"a p f"`

All three examples specify that the report should include only information about backup operations that are active or pending, or that have failed.

To restrict the report to information about backup operations invoked by specified users, type

```
        bkstatus -u users
```

where *users* is a list of user logins that are separated by commas or blank spaces (items separated by blanks are enclosed in double quotes). The report will not include operations that were completed or failed.

By default, only one week's worth of backup status information is saved in this table. If you want backup status information to be saved for more than one week, type

```
        bkstatus -p n
```

where *n* is the number of weeks for which you want information to be saved. You may find it useful to save status information for longer periods, such as a month, so you can examine patterns of servicing particular backup jobs.

Controlling Jobs in Progress

There may be times when you want to suspend a job, cancel a job, or resume a suspended job. You can request these actions by using the `backup` command options `-S`, `-C`, and `-R` (respectively), followed by the appropriate job ID. For example, suppose the backup job with the job ID `back-3288` is in progress when you need the destination device for another purpose. Suspend the job by invoking

```
backup -S -j back-3288
```

Entering `backup -S` without a job ID suspends all outstanding backup operations that were begun by the user entering this command.

Whenever the backup service receives a suspend request, it remembers the destination device currently in use, rewinds the destination device (if appropriate), and yields control of it. When the destination device becomes available for the backup again, you can resume the job by invoking

```
backup -R -j back-3288
```

Entering `backup -R` without a job ID resumes all outstanding backup operations that were begun by the user entering this command. The backup service revalidates the volume label and begins the backup from the beginning of the current volume, not from the point where the suspend request was received.

To cancel the backup job begun in the example above, type

```
backup -C -j back-3288
```

Entering `backup -C` without a job ID cancels all outstanding backup operations that were begun by the user entering this command.

In this case, the backup service rewinds the destination medium currently in use, and yields control of the destination device. In addition, the status display will reflect that this backup job has been canceled.

Displaying the Backup History Log

The `backup` command automatically records all backup operations that have been completed successfully in a backup history log (`/etc/bkup/bkhist.tab`). You can examine the contents of this log through the `bkhistory` command. When invoked without any options, `bkhistory` displays a summary of the contents of the backup history log that includes the following information:

`tag`	A unique identifier associated with a backup operation
`date`	The date and time when a backup operation was performed
`method`	The backup method used for the backup (`incfile, ffile, fimage, fdp, or fdisk`) and whether a migration was requested
`destination`	The name of the device that received the backup archive
`dmname`	The labels of the volumes that received the backup archive
`vols`	The number of volumes required to hold the entire backup archive
`TOC`	Whether a backup archive contains a table of contents and, if so, in what form. The four possible values of TOC are

`online`	The TOC is present online (on the hard disk).
`Arch`	The TOC is present in a volume on a destination device (a removable medium).
`Both`	The TOC is present both online and on a destination device (a removable medium).
`None`	No TOC exists.

Backups are listed alphabetically by operation tag. When a backup operation has been performed more than once during the period reported in the display, the most recent backup is listed first. The following is a sample display of information from a backup history log:

Figure 3-1: Sample Display of a Backup History Log

```
Tag      Date          Method   Destination Dlabels  Vols  TOC
-----------------------------------------------------------------
rootdai  Feb24 12:26   incfile  /usr2/tmp/m cpio1,   2     Online
         1989                   nt          cpio3
rootsun  Feb24 12:31   ffile    diskette    !cpio1,  4     Archive
         1989                               rep1,
                                            !cpio3,
                                            !rep4
usr2dai  Mar02 23:41   ffile    diskette    med1,    3     None
         1989                               med6
usrdai   Jan28 20:29   incfile  file        med1,    3     Archive
         1990                               med2,
                                            med3
```

Note that some entries in the `Dlabel` field begin with the `!` character. This shows that the volume listed has been reused.

Customizing the History Log Display

The `bkhistory` command allows you to customize both the contents and the format of a display. This section explains how to do both.

Customizing the Contents of the Display

The default display contains full details about completed backup operations. You can restrict the type of information that is displayed by using the `-d`, `-t`, or `-o` options to the `bkhistory` command. To restrict a display to information about backups performed on specified dates, enter

```
bkhistory -d dates
```

where *dates* is a list of one or more dates separated by commas. The dates must conform to the syntax used with the `date(1)` command, with one exception: the only argument required is *month.*

To restrict a display to backup operations with specified tags, enter

```
        bkhistory -t tags
```

To restrict a display to backup operations with specified originating devices, enter

```
        bkhistory -o orig
```

where *orig* is in the following format: *oname*:*odevice*: [*omname*].

These options can be combined, as shown in the following example:

```
        bkhistory -d 01,02,03 -o "/usr:/dev/rdsk/c1d0s2 \
        /back:/dev/rdsk/c1d0s8"
```

This command line restricts the display to backup operations completed in January, February, and March from two originating devices: `/usr:/dev/rdsk/c1d0s2` and `/back:/dev/rdsk/c1d0s8`.

Customizing the Format of the Display

In the default display, each field is labeled by a header (such as `Tag`) and has a specified length. Entries that exceed the designated field length wrap to the next line within the field.

You can change the format of the display by using one or more of the following options: `-f`, `-h`, or `-l`. The `-h` option suppresses the headers in the display. This option is useful when the contents of the display are to be filtered by another process, such as an editor.

The `-f` option allows you to suppress field wrap and specify a character for separating fields in your display. This option cannot be used without the `-h` option. To invoke this option, type

```
        bkhistory -h -f c
```

where the value of *c* is the character that will appear as the field separator. The fields in the display appear together in one line. For example, to display the output shown in Figure 3-1 in this format, type

```
    bkhistory -h -f
```

The following display will appear:

```
rootdai;Feb 24 12:26 1990;incfile;/usr2/tmp/mnt;cpio1,cpio3,2;Online
rootsun;Feb 24 12:31 1990;ffile;diskette;!cpio1,rep1,!cpio3,!rep4;4;Archive
usr2dai;Mar 02 23:41 1990;ffile;diskette;med1;3;None
usrdai;Feb 24 12:26 1990;incfile;file;med4,med6;3;Both
usrdai;Jan 28 20:29 1990;incfile;file;med1,med2,med3;3;Archive
```

For clarity, when selecting a separator, do not choose a character that is likely to appear in a field. For example, do not use a colon as a field separator if the display will contain dates in which a colon is used to separate hours from minutes.

To produce the long form of the display, type

```
    bkhistory -l
```

The long form includes the information shown in Figure 3-1, along with the information produced by the `ls -l` command. A display produced in this format looks like the following example:

```
Tag       Dlabel   File information

rootdai   cpio1    -r--r--r--    1 root  sys    2898   Mar 30 11:42  /etc/passwd
rootdai   cpio1    -rw-r--r--    1 root  sys    329    Jan 17 16:05  /etc/group
rootdai   cpio1    -rwxr--r--    1 root  other  646452 Mar 29 12:10  /unix
rootdai   cpio3    -rwxr-xr-x    1 bin   bin    33746  Mar 07 1987   /lib/cc
rootdai   cpio3    -rw-rw-r--    1 root  other  18616  Mar 29 12:10  /lib/libld.a
rootdai   cpio3    -rwsr-xr-x    1 root  sys    11242  Oct 09 17:30  /bin/newgrp
usrdai    med4     -rwxr-xr-x    1 root  other  851    Mar 29 12:10  /usr/oam/bin/add
usrdai    med4     -rwxr-xr-x    1 root  other  759    Mar 29 11:46  /usr/oam/bin/res
```

The entries in this display have values in the `File information` field because the `-l` option to the `bkreg` command was specified for the operations described.

Truncating the Backup History Log

Without some type of control, the backup history log would grow without bounds as more and more backups were completed. Therefore, by default, the system removes any entries that are older than one week. You can override this default and save history information for additional weeks by invoking

```
bkhistory -p period
```

where *period* is the number of weeks for which you want information to be saved in the backup history log. Keep in mind, however, the longer the history log, the greater the number of automatic restore operations that can be done.

Quick Reference to the Backup Service

- Adding an entry to a backup table:

  `bkreg -a` *tag*

  where *tag* identifies a backup operation. The `-a` option must be followed by the `-o`, `-c`, `-m`, and `-d` options and, if you choose, any of the following options that are not required: `-b`, `-t`, `-P`, and `-D`.

- Adding files to a custom exception list for incremental backups:

  `bkexcept -t` *filename* `-a` *pattern* . . .

  where *filename* is the full pathname of the custom exception list, and *pattern* is a list of files and/or sets of files specified by the shell special characters *, ?, or [] that are comma-separated or blank-separated and enclosed in quotes.

- Adding files to the system-supplied exception list for incremental backups:

  `bkexcept -a` *pattern* . . .

  where *pattern* is a list of files and/or sets of files specified by the shell special characters *, ?, or [] that are comma-separated or blank separated and enclosed in quotes.

- Checking the status of backup jobs:

  The status of a backup operation is shown by one of the following keyletters: `p` (pending), `a` (active), `w` (waiting), `s` (suspended), `f` (failed), `c` (completed)

- Displaying the status of backup operations that have either failed or been completed:

  `bkstatus -a`

- Interrupting a backup job:

  `backup -S|-C|-R [-j` *jobid*`] [-u` *users*`] [-A]`

  where `-S` suspends the backup job with the specified *jobid,* `-C` cancels the backup job with the specified *jobid* or cancels the backup job issued by *users* with the specified logins, `-R` resumes the backup job with the specified *jobid*, and `-A` suspends, resumes, or cancels all running backup jobs.

- Defining the number of weeks of backup status information that will be saved:

  ```
  bkstatus -p n
  ```

  where *n* is the number of weeks for which information is to be saved.

- Defining and/or limiting the backup operations to be invoked:

  ```
        backup -t table -o orig -c week:day|demand -m user
  ```

  where *table* is the complete pathname of a custom backup table, and *orig* is the list of originating objects from which backups are to be made. *orig* must be of the form *oname*:*odevice*: [*omname*]. *week* is an integer specifying the week and *day* is an integer or character string specifying the day of the rotation period on which backups will be performed. *user* is the name of the user who is to be notified by `mail` when the backup job is complete.

- Displaying the contents of a backup table:

  ```
  bkreg -C fields|-A|-O|-R -t table
  ```

  where `-C` produces a summary display of the specified *fields*, `-A` displays all fields, `-O` displays a summary of all originating objects, and `-R` displays a summary of all destination devices. `-t` *table* is the name of the backup table.

- Editing an existing entry in a backup table:

  ```
  bkreg -e tag
  ```

  where *tag* identifies a backup operation. If any of the options `-b`, `-c`, `-d`, `-m`, `-o`, `-D`, or `-P` are present, they replace the current settings for the specified entry in the table.

- Displaying the contents of the backup history log:

  ```
  bkhistory
  ```

- Invoking backup operations that are run only on demand:

  ```
  backup -c demand
  ```

- Limiting the growth of the backup history log:

  `bkhistory -p` *period*

  where *period* is the number of weeks for which information will be saved.

- Previewing backup operations:

  `backup -n -e -c` *week:day*|`demand`

  where `-n` alone displays the current day's backup operations, `-e` estimates the number of destination device volumes required, and `-c` *week:day*|`demand` specifies the week and day of the rotation period (or the demand operations) to be previewed.

- Removing an entry from a backup table:

  `bkreg -r` *tag* `-t` *table*

- Removing files to a custom exception list for incremental backups:

  `bkexcept -t` *filename* `-r` *pattern* . . .

  where *filename* is the full pathname of the custom exception list, and *pattern* is a list of files and/or sets of files specified by the shell special characters *, ?, or [] and are comma-separated or blank-separated and enclosed in quotes.

- Removing files from the system-supplied exception list for incremental backups:

  `bkexcept -r` *pattern* . . .

  where *pattern* is a list of files and/or sets of files (specified by the shell special characters *, ?, and [ ]). *pattern* must match an entry in the exception list exactly.

- Requesting the long form of the backup history display:

  `bkhistory -l`

- Restricting the status information that is displayed:

  `bkstatus [-j` *jobids*`] [-s` *states*`] [-u` *users*`]`

  where *jobids* is a list of job IDs, *states* is a list of keyletters representing operation status, and *users* is a list of user login names.

- Selecting an operator mode while invoking the current day's backup operations:

  ```
  backup [-a|-i]
  ```

  where the default system response (in the absence of options) is to send a `mail` message to the operator when a backup operation needs assistance; `-i` prompts the operator at the terminal, and `-a` assumes no operator is present and fails any operations requiring assistance.

- Setting a rotation period for a backup table:

  `bkreg -p` *period* `-w` *cweek* `-t` *table*

  where *period* is the number of weeks in the rotation period, *cweek* is the current week of the rotation period, and *table* is the name of a backup table if a custom backup table is to be used

- Servicing backup operations:

  ```
  bkoper
  ```

  initiates an interactive session by displaying a list of backup operations

- Storing a table of contents online only:

  Tables of contents are stored online by default.

- Storing a table of contents on removable destination devices only, by specifying both `-t` and `-s`:

  `bkreg -m` *method* `-b "-t -s"`

- Storing a table of contents both online and on removable destination devices by specifying `-t`:

  `bkreg -m` *method* `-b -t`

- Storing a table of contents neither online nor on removable destination devices by specifying `-s`:

  `bkreg -m` *method* `-b -s`

- Tailoring the display of the contents of the backup history log:

  `bkhistory -d` *dates* `-o` *orig* `-t` *tags*

  where *dates* is a list of dates that restricts the report to backup operations performed on the specified dates, *orig* restricts the report to the specified originating devices, and *tags* is a list of operation tags.

- Translating an exception list from `ed` syntax to `cpio` format:

  `bkexcept -C` *old_file* > *new_file*

  where *old_file* is the filename of the exception list in `ed` command syntax, and *new_file* is a temporary file that you edit before giving it to `/etc/bkup/bkexcept.tab` for input. After editing the file you can enter `bkexcept -a -` < *new_file*.

- Validating the contents of a custom backup table:

  `backup -n -t` *table*

  where *table* is the name of a custom backup table.

4. DIAGNOSTICS

4 Diagnostics

Introduction

This chapter tells you how to perform diagnostic tests (to identify problems on your system) and how to handle bad blocks on your hard disk.

CAUTION Diagnostics should be performed only by experienced system administrators.

Two sets of diagnostic tests are provided with your system: those used to check your hard disk, and those used to check all other hardware. The diagnostics used to locate, report, and repair hard disk errors reside on the hard disk. Diagnostics for identifying other hardware problems reside in non-volatile, random access memory (NVRAM). These hardware diagnostics must be run through the diagnostic monitor program, dgmon, while the system is in firmware state (system state 5). See the "Machine Management" chapter for a description of firmware state and other system states.

CAUTION To find out which diagnostic tests to perform to identify a particular problem, contact your service representative.

Bad block handling and associated recovery procedures are also described in this chapter. The bad block handling feature is controlled by an automated process called the `hdelogger` daemon; the system administrator does not need to invoke it.

To help you perform diagnostic activities, the UNIX system provides a menu interface. To access a menu of diagnostic tasks, type

```
        sysadm diagnostics
```

The following menu will appear on your screen:

```
1       Diagnosing System Errors

diskrepair - Advises about Disk Error Repairs
diskreport - Reports on Disk Errors
```

If you prefer not to use the menu, you can perform these tasks by using shell level commands instead. The following table shows the shell commands and firmware programs that correspond to the tasks on the menu.

Task to Be Performed	`sysadm` Task	Shell Command or Firmware Program
Accessing the diagnostic monitor (firmware program)	n/a	`shutdown -y -i5` `dgmon`
Adding hand-written reports to the disk error queue	n/a	`hdeadd`
Advice on disk error repairs	`diskrepair`	n/a
Generating a hard disk error report	`diskreport`	`hdelogger -f`
Repairing a bad block that caused a system panic	n/a	`shutdown,` `umountall -k,` `hdeadd,` `hdefix -a,` `fsck -fD,` `mount`

The following table shows those commands available from the `dgmon` program (run from system state 5, firmware state) that do not correspond to tasks on the menu.

Task to Be Performed	sysadm Task	dgmon Command
Displaying the Equipped Device Table	n/a	s
Displaying the Diagnostic Phase Table for a specific device	n/a	l *device_name*
Displaying a help menu for dgmon	n/a	h
Leaving the Diagnostic Monitor	n/a	q
Running diagnostic phases:		
Once on all devices in the Equipped Device Table	n/a	dgn
On all of one type of device	n/a	dgn *device_name*
On a specific instance of a device	n/a	dgn *device_name* #
A specific phase	n/a	dgn *device_name* ph=*n*[-*m*]
Multiple repetitions	n/a	dgn *device_name* rep=*n*
Continuously	n/a	dgn *device_name* soak
In the unconditional mode	n/a	dgn *device_name* ucl

Each task listed above is explained fully in this chapter. In addition, the *System Administrator's Reference Manual* and the *User's Reference Manual* provide manual pages for the shell commands.

Diagnostic tests are not the only programs that must be run from firmware state (system state 5). Other such programs are described in the "Machine Management" chapter.

Detailed explanations of the commands and arguments used with the dgmon program are provided later in this chapter under "Accessing the Diagnostic Monitor (dgmon)."

Overview of Diagnostics

Diagnostics programs are sets of tests, or "phases," that help you evaluate and sometimes repair problems with your 3B2 Computer. There are three categories of diagnostic phases: "default" (those that are run automatically), "demand" (those that are run only when you request them), and "interactive" (those that require your participation). In addition, the diagnostics software includes several tools for repairing some problems once they have been identified. You should run diagnostics at regular servicing intervals and whenever your computer sends a failure message to your console terminal. To find out which diagnostic phases you should run for a particular problem, contact your service representative.

Disk Diagnostics

To ensure that disk errors are caught whenever they occur, a daemon process called `hdelogger`, which monitors disk status, runs constantly on your system. Disk errors cannot be repaired, but the UNIX system provides software tools for making usable the block (section) of the disk in which errors are found. These tools, collectively known as the "bad block handling feature," are described in the "Bad Block Handling" section of this chapter.

Hardware Diagnostics

Hardware diagnostics are a set of tests that can help you locate the physical place (such as a specific mechanical area or electronic circuit board) in which a problem has occurred. Records of diagnostic activities are stored in two tables: the Equipped Device Table (EDT) and the Diagnostic Phase Table (DPT). These tables are described below.

NOTE Diagnostic phase numbers are determined by the hardware configuration of the 3B2 Computer. Therefore certain diagnostic phase numbers may not be defined as the same diagnostic tests from machine to machine.

default phases	All test phases identified as default are run automatically on every device listed in the EDT every time the system is powered up. Testing of default phases can also be run manually from firmware state (system state 5) through the diagnostic monitor program (dgmon).

demand phases	All test phases identified as demand are run when requested through the dgmon program. You may specify a particular device or all devices when running demand phases. Testing of these phases is fully automatic once started.
interactive phases	All test phases identified as interactive are run manually through dgmon. These phases require some operator intervention, such as inserting a floppy diskette into the floppy diskette drive or entering data through a keyboard. You may specify a particular device or all devices when running interactive phases. **WARNING** Running some interactive phases can destroy stored data.
Equipped Device Table (EDT)	Lists the devices that can be tested by running diagnostic phases. (For a copy of this list, see "Determining Available Diagnostic Phases" later in this chapter.)
Diagnostic Phase Table (DPT)	Lists the diagnostic phases that can be run for a particular device. (For a copy of this list, see "Determining Available Diagnostic Phases" later in this chapter.)

Suggestions for Diagnostic Activities

When should you run diagnostic phases?

- Run diagnostics periodically as part of your default, ongoing administrative duties. Print out and keep, for reference, the results of these periodic "checkups." Having printed records of diagnostic test results may help you complete any maintenance procedures indicated.

NOTE Because the LP print service is not available when your system is running in firmware state, you must have a printer connected to the console terminal if you want to print the results of diagnostic tests and programs.

- Run diagnostic phases when you receive a system failure message, alerting you to a problem with a disk drive or other system component. This problem may be intermittent or it may cause a complete failure that renders some devices or system services inoperative. To find out which phases to run, contact your service representative.

Accessing the Diagnostic Monitor (dgmon)

The diagnostic monitor program allows you to perform tests and analyses that will help you resolve hardware problems. Each test is called a "phase." You can perform many test phases at different intervals on a device connected to your system.

NOTE The diagnostic monitor program (dgmon) can be executed only when the system is in firmware state (system state 5). Because users can't access a system that's in firmware state, it's a good idea to limit your use of the diagnostic monitor to times when system usage is low and to warn users that the system will be unavailable at those times.

To use the diagnostic monitor, complete the following procedure.

1. Log in at the console terminal as `root`.
2. Change the system to firmware state (system state 5) by issuing the following command:

   ```
   shutdown -y -g0 -i5
   ```

   If any other users are logged in, give them a grace period of at least 300 seconds (`-g300`) in which to log off. (See the "Machine Management" chapter for details.)
3. Enter the firmware password; the default is mcp. (Remember that passwords are not displayed on the screen.)
4. If your computer does not have the standard NVRAM set, either the > or the < prompt appears, followed by an audible beep. (If such a prompt does not appear, skip to the next step.) At either of these prompts, type `boot`.
5. The system displays the following prompt:

   ```
   Enter name of program to execute [unix]:
   ```

   Enter dgmon to run the diagnostic monitor program. (See Step 8 in this procedure for a description of dgmon options.)

NOTE

You can run several programs, such as `filledt` and `edt`, from the firmware state. The use of other programs is described in the "Machine Management" chapter.

6. Enter

   s

   to see the EDT.

7. Choose the device on which you want to run diagnostics from the EDT.

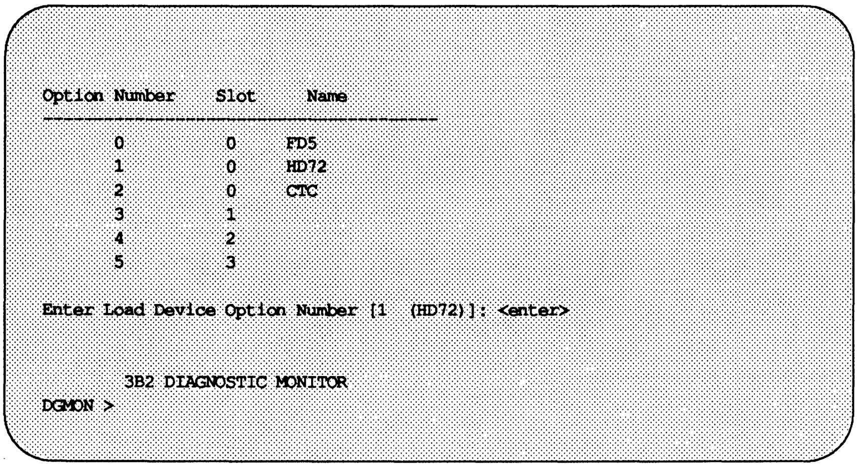
   Figure 4-1 shows the screen activity for the first five steps of this procedure. The message `SELF-CHECK` means that the default diagnostic phase is being run at this time.

Figure 4-1: Accessing the dgmon **Program**

```
# shutdown -y -i5 -g0

Shutdown started.    Wed May 16 17:21:01 MDT 1990

Broadcast Message from root (console) on mach_name Wed May 16 17:21:06...
THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged.

#
INIT: New run level: 5
The system is coming down.  Please wait.
System services are now being stopped.
Print services stopped.
cron aborted: SIGTERM

The system is down.

SELF-CHECK

FIRMWARE MODE
firmware_password

Enter name of program to execute [unix]: dgmon
        Possible load devices are:
```

(continued on next page)

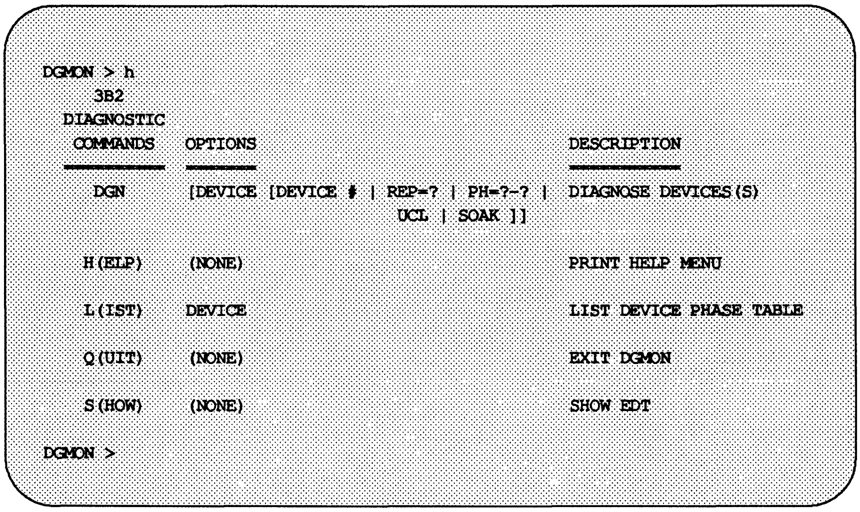
Figure 4-1: Accessing the dgmon Program (continued)



```
Option Number    Slot     Name
----------------------------------------
        0          0     FD5
        1          0     HD72
        2          0     CTC
        3          1
        4          2
        5          3

Enter Load Device Option Number [1  (HD72)]: <enter>


        3B2 DIAGNOSTIC MONITOR
DGMON >
```

8. At the DGMON > prompt, you can issue any of the following commands: `help` (or h), `list` (or l), `quit` (or q), show (or s), and dgn. If you enter h, a menu of commands and options will be displayed, as shown in Figure 4-2.

Figure 4-2: Sample Output of the help Command for dgmon



```
DGMON > h
   3B2
 DIAGNOSTIC
  COMMANDS   OPTIONS                                    DESCRIPTION
 ---------   -------                                    -----------
    DGN      [DEVICE [DEVICE # | REP=? | PH=?-? |       DIAGNOSE DEVICES(S)
                                  UCL | SOAK ]]

   H(ELP)     (NONE)                                    PRINT HELP MENU

   L(IST)    DEVICE                                     LIST DEVICE PHASE TABLE

   Q(UIT)     (NONE)                                    EXIT DGMON

   S(HOW)     (NONE)                                    SHOW EDT

DGMON >
```

The functions performed by the `dgn` commands are described below. You can run the `help`, `list`, `quit`, and `show` commands by entering the first letter of the command name, as shown in parentheses.

`help` (h) Provides a help menu that describes the `dgmon` commands.

`list` (l) Displays a table that lists the diagnostic phases for each configured device. The `l` command without an argument displays the tables for all configured devices.

To display a list of diagnostic phases for a specific device, use the name of the device as an argument to this command. For example, to examine the diagnostic phases for the system board (device name of `sbd`), type

```
l sbd
```

`quit` (q) Exits from the `dgmon` program.

`show (s)`	Displays the Equipped Device Table (EDT), which lists the devices that can be tested by running the diagnostic phases.
`dgn`	Runs the diagnostic phases. By default, `dgn` runs the default phase once on the device specified on the command line. To run other phases, you must include ph=*n*[−*m*] on the command line. Optional arguments to the `dgn` command are described below.

device_name	Runs diagnostic phases on a configured device type. For example, when you issue the command `dgn sbd`, all the default diagnostic phases are run on the system board.
device_name #	Runs diagnostic phases on a particular configured device. For example, the command `dgn ports 0` runs all the default diagnostic phases on the first ports board.
`rep=`*n*	Runs phases for a specific number of times. Valid numbers range from 1 to 65536.
ph=*n*[−*m*]	Runs a specific phase or string of phases where *n* is the number of a specific phase and *n*−*m* are numbers defining a range of phases. When running specific phases, be sure you know which phase you want because some phases can destroy stored data. When possible, run interactive phases separately. If you do not specify a range of phases in the command line, only the default phases are run.
`ucl`	Runs the phases in the unconditional mode. In this mode, testing runs to completion, even when a phase fails. Results are displayed as each phase is completed. This mode cannot be used with the `soak` option.
`soak`	Runs the phases continuously, and stores the results until testing is completed. This allows you

to check for intermittent problems by comparing the number of failures against the number of times the phase ran.

For each specified device, soak runs all default and demand phases in sequence within the requested range of phases. To stop the soak option, enter a character at the console or use the rep option.

The soak option cannot be used with interactive phases.

9. To leave the diagnostic monitor, type q. The system returns you to firmware state and redisplays the following firmware prompt:

```
Enter name of program to execute [unix]:
```

10. To reboot the system, run the unix command from the hard disk containing the root partition.

Running Diagnostic Phases

WARNING

Before using the diagnostic monitor, contact your service representative. Run diagnostic phases only under specific instructions. Running some interactive phases can destroy stored data.

While in the diagnostic monitor, you can run diagnostics for the entire computer or for a part of the computer. For example, you may want to test all the devices of one type (such as all the ports boards), a specific device (such as the system board), or a particular phase or range of phases for a device or type of device.

NOTE

If the Equipped Device Table (EDT) is corrupted, don't run the `dgmon` program until you have run the `filledt` program from firmware state (system state 5). The `filledt` program initializes the EDT which is read by `dgmon` as its active configuration.

This section explains how to determine which phases are available for testing on your 3B2/400 Computer. It also recommends a sequence in which to run diagnostic phases. You can use this sequence or, once you know the purpose of each phase, you can develop your own sequence. The important point is that you avoid running phases in random order. Because some phases may cause data to be lost, running phases in random order could jeopardize your data.

Determining Available Diagnostic Phases

1. In firmware state (system state 5), access the diagnostic monitor by entering the `dgmon` command; the `DGMON >` prompt will appear on your screen.

2. At the prompt, type `s` to display a list of devices on which diagnostics may be performed.

   The command displays information about only one device at a time. After you have examined the information about the first device and are ready to display the information about the next one, you must signal the command that you're ready for the next display. To do so, press a key (any key) on your keyboard. Figure 4-3 shows sample output of the `s` (`show`) command.

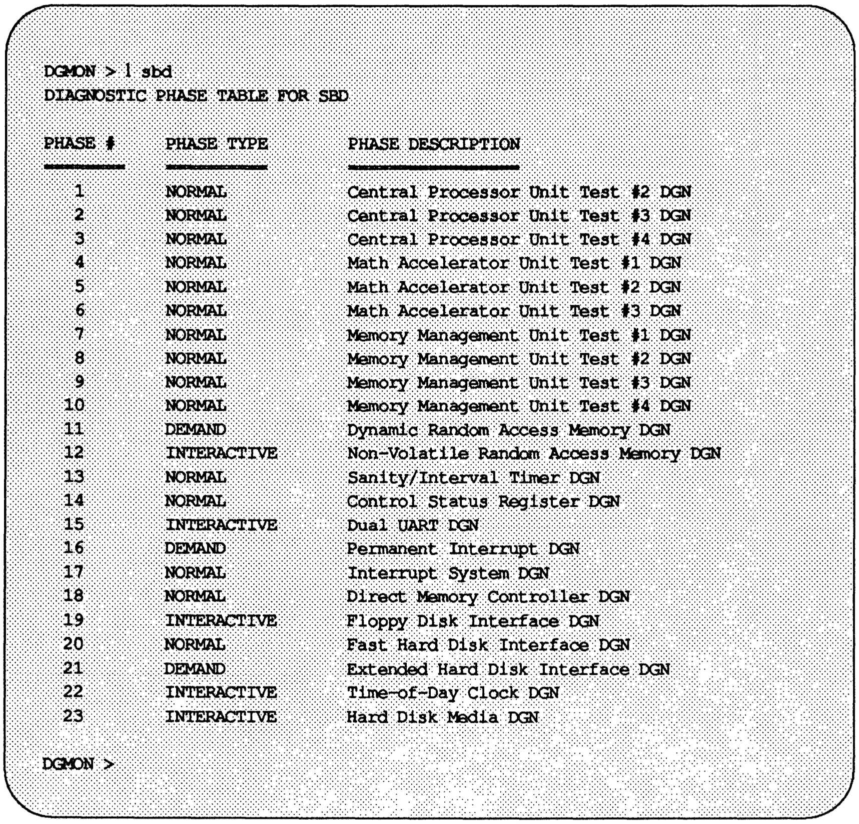
Figure 4-3: A Sample Display of Devices that Can Be Tested with the dgn Command

```
DGMON > s

Current System Configuration

System Board Memory size: 4 megabyte(s)

00 - device name = SBD      , occurrence = 0, slot = 00, ID code = 0x02
     boot device = y, board width = double, word width = 2 byte(s),
     req Q size = 0x00, comp Q size = 0x00, console ability = y,
     pump file = n
     subdevice(s)
     #00 = FDS      , ID code = 0x01, #01 = HD72      , ID code = 0x05
     #02 = HD30      , ID code = 0x03

Press any key to continue

01 - device name = PORTS    , occurrence = 0, slot = 01, ID code = 0x03
     boot device = n, board width = single, word width = 2 byte(s),
     req Q size = 0x03, comp Q size = 0x23, console ability = y,
     pump file = y

Press any key to continue

02 - device name = CTC      , occurrence = 0, slot = 02, ID code = 0x05
     boot device = y, board width = single, word width = 2 byte(s),
     req Q size = 0x10, comp Q size = 0x20, console ability = n,
     subdevice(s)
     #00 =           , ID code = 0x00, #01 =           , ID code = 0x00

DONE

DGMON >
```

This sample display lists three devices on which you can perform diagnostics: `sbd`, `ports`, and `ctc`. These device names are also used as arguments to the dgn command. (See "Recommended Sequence for Running Phases" later in this section.)

3. At the prompt, type `l` *device_name* to display a list of phases that may be run on a given device (that is, the contents of the diagnostic phase table). Figure 4-4 shows a sample display of the phases that can be run on the `sbd` device (the system board).

Figure 4-4: A Sample Display of Diagnostic Phases for the System Board



```
DGMON > l sbd
DIAGNOSTIC PHASE TABLE FOR SBD

PHASE #      PHASE TYPE         PHASE DESCRIPTION
-------      ----------         -----------------
   1         NORMAL             Central Processor Unit Test #2 DGN
   2         NORMAL             Central Processor Unit Test #3 DGN
   3         NORMAL             Central Processor Unit Test #4 DGN
   4         NORMAL             Math Accelerator Unit Test #1 DGN
   5         NORMAL             Math Accelerator Unit Test #2 DGN
   6         NORMAL             Math Accelerator Unit Test #3 DGN
   7         NORMAL             Memory Management Unit Test #1 DGN
   8         NORMAL             Memory Management Unit Test #2 DGN
   9         NORMAL             Memory Management Unit Test #3 DGN
  10         NORMAL             Memory Management Unit Test #4 DGN
  11         DEMAND             Dynamic Random Access Memory DGN
  12         INTERACTIVE        Non-Volatile Random Access Memory DGN
  13         NORMAL             Sanity/Interval Timer DGN
  14         NORMAL             Control Status Register DGN
  15         INTERACTIVE        Dual UART DGN
  16         DEMAND             Permanent Interrupt DGN
  17         NORMAL             Interrupt System DGN
  18         NORMAL             Direct Memory Controller DGN
  19         INTERACTIVE        Floppy Disk Interface DGN
  20         NORMAL             Fast Hard Disk Interface DGN
  21         DEMAND             Extended Hard Disk Interface DGN
  22         INTERACTIVE        Time-of-Day Clock DGN
  23         INTERACTIVE        Hard Disk Media DGN

DGMON >
```

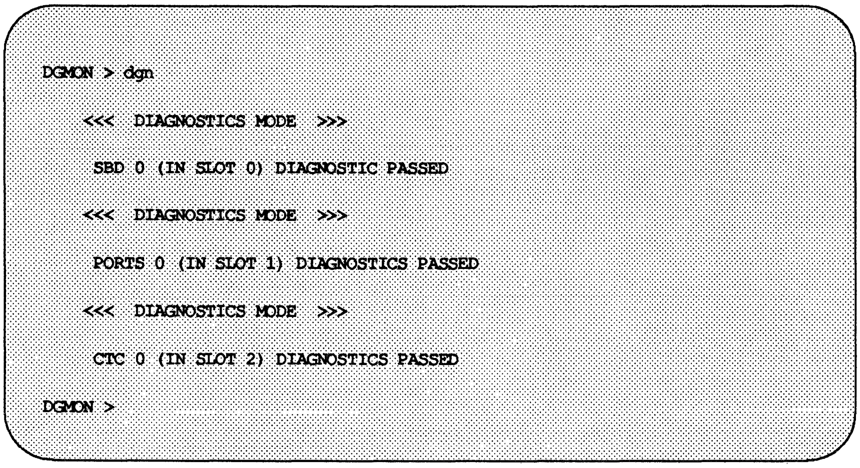
You can display a list of diagnostic phases for the remaining devices by using the other device names found in Step 2, above, as arguments to the `l` command (such as `l ports` or `l ctc`). Phase tables similar to the one shown in Figure 4-4 will be generated for each of these devices.

Recommended Sequence for Running Phases

NOTE When running diagnostic phases, be sure to print out the results on your console printer. Discuss these results with your service representative when requesting a service call.

1. In firmware state (system state 5), access the diagnostic monitor program by running the dgmon command; the DGMON > prompt will appear on your screen.
2. To list the devices in the equipped device table (EDT), type s at the prompt. (See the previous section for details.) The devices shown in this table can be tested.
3. To locate a device that's causing trouble, run the dgn command with no options. Figure 4-5 shows a typical display for a healthy machine.

Figure 4-5: A Sample Display of the dgn Command



```
DGMON > dgn

    <<<  DIAGNOSTICS MODE  >>>

     SBD 0 (IN SLOT 0) DIAGNOSTIC PASSED

    <<<  DIAGNOSTICS MODE  >>>

     PORTS 0 (IN SLOT 1) DIAGNOSTICS PASSED

    <<<  DIAGNOSTICS MODE  >>>

     CTC 0 (IN SLOT 2) DIAGNOSTICS PASSED

DGMON >
```

4. If a test of a particular device results in the display of the error message DIAGNOSTICS FAILED, you should focus testing on this device. Run all

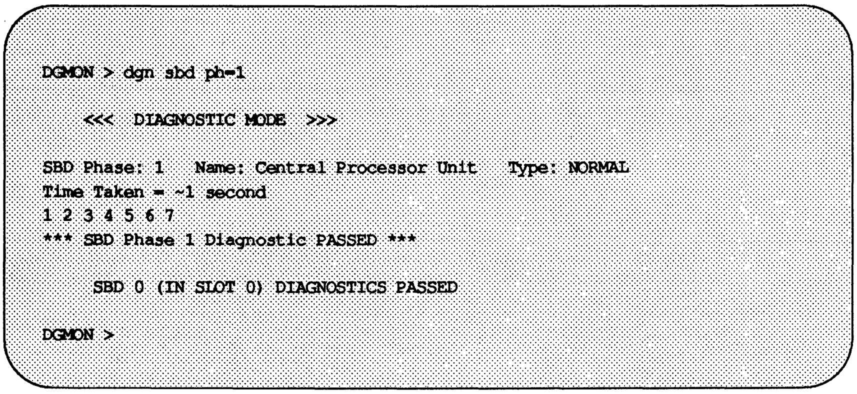
default phases on the device that failed with the following command:

dgn *device_type*

where *device_type* is the type of device specified by the error message. For example, if a failure occurred on the system board, the device type would be sbd.

Figure 4-6 shows an example of a diagnostic test: the phase 1 test of the system board (sbd).

Figure 4-6: A Sample Display of the Phase 1 Test of the System Board



```
DGMON > dgn sbd ph=1

    <<< DIAGNOSTIC MODE >>>

SBD Phase: 1   Name: Central Processor Unit   Type: NORMAL
Time Taken = ~1 second
1 2 3 4 5 6 7
*** SBD Phase 1 Diagnostic PASSED ***

     SBD 0 (IN SLOT 0) DIAGNOSTICS PASSED

DGMON >
```

5. If the phase fails again, another error message appears on your screen. In this case, you should request numerous repetitions of the test with either the soak or the ucl option. The soak option runs both default and demand phases. The ucl option runs all phases to completion, regardless of whether errors are detected.

   You can request repeated runs of a particular phase with either of these options by assigning a value to the rep ("repetitions") argument, such as rep=10, if you want a phase to be run ten times.

   For example, to test the system board ten times with the soak option, type

   ```
   dgn sdb soak rep=10
   ```

6. If you are testing because of a system failure or intermittent trouble, and all the default phases pass, run the demand phases next. To find out which phases are demand phases, refer to your results from Step 2, above. For example, Figure 4-4 shows that the demand tests for the system board are phases `11, 16,` and `21`. To perform these phases, type

   ```
   dgn sdb ph=11,16,21
   ```

7. If any of the demand phases fail, run them again with the `soak` or `ucl` option, specifying the desired number of repetitions. If any of the demand phases fail, another error message will appear on your screen. In this case, you should request repetitions of this device test by setting the number of repetitions to ten (`rep=10`) with either the `soak` or `ucl` option. The `soak` option runs both the default and demand phases. The `ucl` option completes all specified phases even when errors are detected.

   For example, to perform the demand phases on the system board ten times with the `soak` option, type

   ```
   dgn sdb soak ph=11,16,21 rep=10
   ```

8. If failures have occurred while phases are being run with the `soak` option or `ucl` option, omit these failed phases from the phase list and continue running the other phases.

   For example, suppose you are testing the system board (as described in Step 7) and phase 16 fails. To exclude phase 16 from the demand tests, type

   ```
   dgn sdb soak ph=11,21 rep=10
   ```

9. When one phase fails, the phases that follow it may not be run. For example, if you specify `ph=1-9` and a failure occurs on phase 4, phases 5 through 9 will not be run. If any devices in the EDT are not tested because of this type of premature test termination, those devices should be tested by running a phase with the specific device number listed in this table. To continue the example described above, after the failure in phase 4, testing stops. Restart the same type of test but specify `ph=5-9` to ensure that the remaining phases are done (or use the `ucl` option).

10. After running all default phases and demand phases, run those interactive phases recommended by your service representative. (To obtain the interactive phase numbers for a device, run the 1 *device_name* command.)
11. After you have run all diagnostic phases, discuss the results with your service representative.

Examples of dgn Commands

This section provides some examples of valid dgn commands using various options. For a complete explanation of the dgn command, see dgmon(8) in the *System Administrator's Reference Manual*.

`dgn`	Runs all default phases once on the devices listed in the EDT. The results for each test phase are displayed as that phase is complete. If any of the phases fail, testing stops and a failure message appears.
`dgn sbd 0`	Runs all default phases once on the system board. The results for each test phase are displayed as that phase is complete. If any of the phases fail, testing stops and a failure message appears.
`dgn ports`	Runs all default phases once on all the ports boards. If any of the phases fail, testing stops and a failure message appears.
`dgn ports 1`	Runs all default phases once on ports board 1. The results for each test phase are displayed as each phase is complete. If any of the phases fail, testing stops and a failure message appears.
`dgn ports 2 ucl`	Runs all default phases once on ports board 2. The results for each test phase are displayed as each phase is complete. If any of the phases fail, testing stops and a failure message appears.
`dgn sbd ph=3`	Runs phase 3 once on the system board. The results appear as the phase is completed.
`dgn sbd ph=1-4`	Runs phases 1 through 4 once on the system board or until a failure occurs. The results of each phase appear as that phase is completed.
`dgn sbd rep=3 ph=3`	Runs phase 3 three times. If any part of the phase fails, testing stops and a failure message appears.
`dgn ucl`	Runs all default phases once on every device listed in the Equipped Device Table. The results of each phase appear as that phase is complete. If a phase fails, an error message appears and testing continues.

`dgn ucl rep=3`	Runs all default phases three times. The results of each phase appear as that phase is complete. If a phase fails, an error message appears and testing continues.
`dgn soak`	Runs all default and demand phases on all boards until you enter a character at the console. When testing stops, the results appear.
`dgn ports 1 soak`	Runs all default and demand phases on ports board 1 until you enter a character at the console. When testing stops, the results appear.
`dgn sbd soak ph=1-3`	Runs phases 1 through 3 until a character is entered on the console terminal. When testing stops, the results appear.
`dgn sbd soak rep=10 ph=11`	Runs phase 11 ten times and then displays a summary of the results. If a phase fails, an error message appears and testing continues.
`dgn soak rep=25`	Runs all default and demand phases on all boards 25 times and displays the results when testing is completed. If a phase fails, an error message appears and testing continues.

The following samples show what you can expect when running the three types of diagnostic phases.

Sample Default Diagnostic Phase

NOTE Diagnostic phase numbers are determined by the hardware configuration of your 3B2 Computer. Therefore, certain diagnostic phase numbers may not be defined as the same diagnostic test from machine to machine.

CPU Test #2 is a default phase run on the system board; it takes about one second to run. The following command line and system responses show the successful execution of this test.

```
DGMON > dgn sbd ph=1

     <<<  DIAGNOSTIC MODE  >>>

SBD Phase: 1   Name: Central Processor Unit:  Type: NORMAL
Time Taken = ~1 seconds
1 2 3 4 5 6 7
*** SBD Phase 1 Diagnostic PASSED ***

     SBD 0 (IN SLOT 0) DIAGNOSTICS PASSED
```

Sample Demand Diagnostic Phase

The Permanent Interrupt Diagnostic Phase is a demand phase that is run on the system board. This phase takes about one second to run. The following command line and system responses show the successful execution of this test.

CAUTION A failure of this phase may affect those phases that assume the system is clear of interrupts. Do not trust the results of these other phases. Contact your service representative.

```
DGMON > dgn sbd ph=11

    <<<  DIAGNOSTIC MODE  >>>

SBD Phase: 11   Name: Dynamic Random Access Memory:  Type: DEMAND
Time Taken = ~ 11 minutes
1 2 3 4
*** SBD Phase 11 Diagnostic PASSED ***

    SBD 0 (IN SLOT 0) DIAGNOSTICS PASSED

DGMON >
```

Sample Interactive Diagnostic Phase

The Non-Volatile Memory Diagnostic Phase is an interactive phase run on the system board. This phase takes about one second to run. The following command line and system responses show the successful execution of this test.

```
DGMON > dgn sbd ph=12

    <<<  DIAGNOSTIC MODE  >>>

SBD Phase: 12   Name: Non-volatile Random Access Memory:  Type: INTERACTIVE
Time Taken = ~1 seconds

WARNING: This test can destroy Non_volatile RAM contents !
Are you certain you wish to execute this diagnostic [y/n]: y
*** SBD Phase 1 Diagnostic PASSED ***

     SBD 0 (IN SLOT 0) DIAGNOSTICS PASSED

DGMON >
```

Recovering from System Trouble

This section provides instructions for handling hardware problems and software errors and describes how to return the system to a usable state.

A system experiencing trouble may be in any of the following conditions:

- The system is running, but throughput is degraded.
- The system is running after an automatic reboot.
- The system is halted or in an unknown state.

NOTE You must log in as `root` at the console terminal before proceeding.

Identifying System Trouble

1. Consult the information about troubleshooting in your computer installation manual for problems that occurred during the first-time setup of your computer. If you need additional information, call your service representative.
2. If the operating system is running, use the `/usr/sbin/errdump` command to display the error history file, which includes the contents of various system registers and the last five error messages received. Enter

   ```
   errdump
   ```

   If possible, send the output of this command to a printer, as follows:

   ```
   errdump | pr | lp
   ```

Example of `errdump`

Use the `errdump` command when there is system trouble but the operating system is still running. The following command lines and system responses show how to run `errdump`.

```
# errdump
nvram status:   sane

csr:     0x0258  (unassigned) (clock) (pir9) (uart)

psw:     rsvd CSH_F_D QIE CSH_D OE NZVC TE IPL CM PM R I ISC TM FT
(hex)      0        0   0     0  0    0  0   f  0  0 1 0   5  0  3

r3:      0xffffffff
r4:      0x400d554c
r5:      0x866f6300
r6:      0xc002001e
r7:      0x0021413f
r8:      0x866f62cc
oap:     0x400806e0
opc:     0x40010d3f
osp:     0x40080708
ofp:     0x40080708
isp:     0x40080004
pcbp:    0x40041a9c

fltar:   0x866f62cc
fltcr:   reqacc  xlevel  ftype
         0xb     0x0     0x3

         srama           sramb
[0]      0x02083000      0x0000011f
[1]      0x02083900      0x00000030
[2]      0x0209b860      0x00000074
[3]      0x0209bc00      0x00000015

         Panic log

[0]      Thu Oct 20 07:37:11 1988
         KERNEL MMU FAULT (F_STDLEN)

[1]      Sun Oct 23 15:43:59 1988
         SYSTEM BUS TIME OUT INTERRUPT

[2]      Wed Nov  2 15:47:50 1988
         KERNEL BUS TIMEOUT

[3]      Sat Dec 31 18:59:59 1988
         D,LxTV

[4]      (0xffffffff,0xffffffff,0xffffffff,0xffffffff)
#
```

Performing a System Dump

You should perform a system dump when a `SYSTEM FAILURE` message appears on the screen of the console terminal. To perform a system dump from firmware state (system state 5), complete the following procedure:

1. If the operating system crashes, it automatically enters firmware state (system state 5). A `SYSTEM FAILURE` message is displayed.
2. After the `SYSTEM FAILURE` message appears, the following prompt appears:

   ```
   FIRMWARE MODE
   ```

   After this prompt, enter the firmware password (the default password is `mcp`). When you see the prompt

   ```
   Enter name of program to execute [unix]:
   ```

   enter `sysdump`. This program dumps the system core image to diskettes. Because it runs in system state 5, it does not depend on the integrity of the root file system.
3. Gather the appropriate number of formatted diskettes for the dump. Use this rule of thumb: you need one diskette for each 0.75 megabytes of memory in your configuration.

With...	Use...
1 megabyte	2 diskettes
2 megabytes	3 diskettes
3 megabytes	4 diskettes
4 megabytes	6 diskettes

   Allow about 4.5 minutes per diskette.
4. Follow the instructions displayed on the screen, which prompt you to insert diskettes into the diskette drive.
5. When the dump is finished, the program exits and gives you the option of executing another program.

   ```
   Enter name of program to execute [unix]:
   ```

NOTE You may run diagnostics after you do the dump. Before you can run diagnostics from the hard disk, however, the root file system must be undamaged. Refer to "Accessing the Diagnostic Monitor" earlier in this chapter.

6. To return to multi-user state (system state 2) execute the boot program. When you see the prompt

   ```
   Enter name of program to execute [unix]:
   ```

   enter

   ```
   /stand/unix
   ```

   In general, once you have performed the system dump, be sure to boot your system by running the `/stand/unix` program. If you boot your system by running the `/stand/system` program, the resulting `/stand/unix` that is generated may not match the previous version.

7. It is good practice to make a copy of the current version of `/stand/unix` (the version that was current at the time of the system crash) on an additional diskette. Making this copy of `/stand/unix` allows the dump to be analyzed properly by `/usr/sbin/crash`. To make a copy of `/stand/unix`, use the `cpio` command. (See the "Storage Device Management" chapter for information on how to make a copy of a file system on a removable storage medium, such as a diskette or a cartridge tape.)

   You may want to contact your service representative for help in analyzing a system dump.

Example of `sysdump`

If there is system trouble and the operating system is not running, run the `sysdump` command. The following command line entries and system responses show how to run `sysdump`.

```
SYSTEM FAILURE
<mcp>

Enter name of program to execute [ ]: sysdump

Do you want to dump the system image to the floppy diskette?
Enter 'c' to continue, 'q' to quit: c

Insert first sysdump floppy.
Enter 'c' to continue, 'q' to quit: c

Dumping main store
...............................................................................
.............................................................
If you wish to dump more of main store,
insert new floppy.

Enter 'c' to continue, 'q' to quit: c

Dumping more main store
...............................................................................
.............................................................
If you wish to dump more of main store,
insert new floppy.

Enter 'c' to continue, 'q' to quit: c

Dumping more main store
...............................................................................
............................................
Dump completed.
three floppies written

Returning to firmware

SELF-CHECK

FIRMWARE MODE

Enter name of program to execute [  ]: /stand/unix
```

Bad Block Handling

A disk is a magnetic medium on which digital data (measured in bits and bytes) are stored in logical sections called "blocks." A block usually contains 512, 1024, or 2048 bytes of data and is handled by the UNIX system as a single unit. Because the bit density is very high (millions of bits of data are packed into a small space), the magnetic properties of a disk must be precisely uniform. Variations in uniformity mean that some bit patterns may be stored more easily in one block than in another. This variation in uniformity is normally insignificant. However, when these variations become so great that data can no longer be stored reliably in a particular block, that block is labeled "bad." There are three categories of bad blocks, each of which is described under "Bad Block Recovery" later in this chapter.

If the pattern of data stored in a block coincidentally matches the nonuniformity of the disk, a bad block may escape recognition. However, the nonuniform block will eventually be discovered if the disk remains active. Bad blocks are discovered when a data read or write fails after several successive attempts. Read failures alone do not guarantee the existence of a bad block because they may also be caused by problems in the format of the disk, a failure in the disk controller, or hardware flaws. Write failures generally signal problems with the disk's format or failures in the disk or disk controller hardware.

The bad block handling feature does not distinguish genuine bad blocks from other types of problems; it reports all types of read and write failures, most of which are not caused by bad blocks. (Reports can be viewed by issuing the command `hdelogger -f`.) The distinction is unimportant, however, because all read and write failures indicate problems that must be repaired.

To fix read and write problems, you must reformat the disk or arrange to repair the hardware. In either case, you should call your service representative; especially if several failures occur about the same time.

How the UNIX System Handles Bad Blocks

You cannot really "fix" a bad block; you can only find a way to work around it. One way you can work around a bad block is by using a "media-specific data area."

A media-specific data area is a small portion of a disk that is isolated from the default data area; that is, it is not used by normal UNIX system commands and system calls. This isolated portion of the disk contains a description of the properties of the disk and other media-specific data.

The media-specific data area includes a set of blocks called the surrogate image region. The purpose of this region is to hold the data that should have been stored in a bad block. The media-specific data area also includes a mapping table to map a bad block on the disk to one of the surrogate image blocks. The disk driver software in the operating system has a map showing the original location of the data in the bad block and the new location of that data in the surrogate image block. Data are read from or written to the surrogate image block via the addresses in this mapping table. Address mapping is transparent to software programs accessing the data.

The UNIX system has a software feature called bad block handling. This feature extends the useful life of an integral hard disk by

- Detecting and recording blocks that are no longer usable
- Reminding you that you need to "fix" some bad blocks that have been identified
- Restoring the usability of the disk in spite of the bad blocks that exist

Most disk drives are manufactured with a few defective blocks. Bad blocks detected in the manufacturer's quality control checks are identified on a label on the drive when it is delivered. The bad block handling feature provides special software for recording the known bad blocks and for mapping any additional ones that are encountered. If a surrogate block becomes bad, the software remaps the original bad block to a new surrogate block.

NOTE This feature can be used only with hard disk devices. There is no comparable feature for diskettes or tapes.

New bad blocks will seldom occur if you do not move or vibrate the disk drive while the disk is spinning. When a new bad block does occur, the data stored in the bad block are lost and the disk may be unusable in its current state. The bad block handling feature helps restore the usability of a disk. How much data you lose depends on the adequacy of the backup procedures you have established. (For a discussion of backup procedures, see the "Backup Service" chapter.)

Blocks that Cannot Be Mapped

There are a few special blocks in the isolated portion of the disk that cannot be mapped: the disk block containing the physical description of the disk and the disk block or blocks containing the mapping table. All other blocks, including the surrogate image blocks, can be mapped.

When Are Bad Blocks Detected?

Bad blocks are detected when read or write operations fail for several successive attempts. When these failures occur, data being read or written at the time are lost, but the system can restore the use of the disk by mapping the bad blocks to usable surrogate blocks.

There are several questions that are frequently asked about bad block handling.

- Why doesn't the system try to discover that a given block is bad while the system still has the data in memory?

  Besides the undesirable increase in system size and complexity, severe performance degradation would result. Also, a block can become defective after the copy in memory no longer exists.

- Why doesn't the system periodically test the disk for bad blocks?

  Because there is no reasonable and reliable way to do this kind of testing, reading blocks with their current contents may not reveal a bad block. Even if a thorough bit pattern test could be devised using ordinary read and write operations, it would take so long that you would never run it. The disk manufacturer has tested the disk using extensive bit pattern tests and special hardware. All manufacturing defects have been identified.

- Why are disks with manufacturing defects used?

  By selling disks that contain a modest number of defects, manufacturers can greatly increase their yield and thereby reduce the cost of each disk. Many manufacturers take advantage of this cost reduction to provide a more powerful system at a low price.

Bad Block Recovery

The bad block handling feature provides mechanisms for detecting bad blocks and mapping them to surrogate blocks. The remainder of this section describes three ways that recovery from bad blocks can be handled.

Automatic Recovery from a Bad Block

One of the tasks that the UNIX system performs regularly is a check of the hard disks for bad blocks. (This check is always performed when the system is in multi-user state, that is, in system state 2.)

The UNIX system keeps track of the location of each file on a disk by recording several data about it in a "virtual node" (vnode) table. Two of the data items in this table identify the location of the file (the physical block number on the disk that specifies the sector, head, and cylinder numbers) and its size (the number of data blocks occupied by the file).

The following scenario explains the automatic process of handling bad blocks. Physical block 3 is a surrogate block on the disk for physical block 42. Block 42 is a block in the middle of a text file that is five data-blocks long. Block 3 has become a bad block since the file was last read. Now the file needs to be read again. When block 42 is reached, the driver for the integral disk sees that block 42 is mapped to block 3 and attempts to read block 3. But block 3 is now bad and cannot be read. When the integral disk driver determines that block 3 is unreadable, the following messages are sent to the system console:

```
WARNING: unreadable CRC hard disk error: maj/min = 17/0

        block # = 3

WARNING:
hard disk: cannot access sector 3, head 0, cylinder 0, on drive 0

Disk Error Daemon: successfully logged error for block 3 on disk
maj=17 min=0
```

NOTE CRC stands for Cyclic Redundancy Check, an error checking method. The numbers assigned to `maj` and `min` are the major and minor device numbers of the disk on which the error occurred. (See the "Device Identification Via Special Files" section of the "Storage Device Management" chapter for a complete description of major and minor device numbers.)

The attempt to read this text file has failed. When you notice the message on the console, run the `shutdown` command to change the system to single-user state (system state 1). While `shutdown` is running, the following message is sent to the system console:

```
Disk Error Daemon: Disk maj=17 min=0: 1 errors logged
```

In this scenario, a bad block has been identified, reported, and logged by automated mechanisms in the system. Most bad blocks are handled in this way. The following sections describe how these automated mechanisms work.

Identifying Disks

Disks are identified by their major and minor device numbers (on both single-disk and multi-disk computers). Messages printed by the bad block handling mechanisms use the major and minor numbers rather than any other names. The utilities of the bad block handling feature can be given these numbers as arguments when more specialized operations must be used.

Detecting Bad Blocks

The disk driver detects a bad block when the disk fails to access that block after several attempts. To be sure that the problem is not being caused by the position of a read or write head, the driver repositions the read and write heads between access attempts. When the driver is still unable to access this block, it can safely be assumed that the block is defective.

Reporting and Logging Bad Blocks

When a block is determined to be inaccessible, the disk driver reports the defective block to the bad block logging mechanism which, in turn, notifies the system administrator by sending a message to the system console. For example, in the scenario described above, the following message was sent to the console:

```
WARNING: unreadable CRC hard disk error: maj/min = 17/0

         block # = 3
```

This output is used as input to the hard disk error logger commands. In this case, for example, device 17/0 and block #3 would be arguments to the `hdelogger` commands.

The disk driver also reports the error by displaying the following message on the system console:

```
WARNING: hard disk: cannot access sector 3, head 0, cylinder 0,
on drive 0
```

The logging mechanism then attempts to record the error in the disk error log located in the isolated portion of the disk. If the error is logged successfully, the following message is sent to the system console:

```
Disk Error Daemon: successfully logged error for block 3 on disk
maj=17 min=0
```

The logging mechanism is run by the driver for the hard disk error log (`hde-log`) and by a hard disk error daemon process called `hdelogger` that is, in turn, run by the `/sbin/init` process. The error log driver provides both mechanisms for reports and access to the reserved disk areas needed by the bad block handling feature. (This driver can queue a maximum of 18 reports.) Because one track on a 10-megabyte or 30-megabyte disk has 18 sectors, a bad track can have up to 18 reports, one for each bad sector. The disk error daemon gets reports from this queue and adds each report to the disk error log.

The disk error daemon has another reporting role. When the system state changes (for example, when you turn on your computer, shut it off, or shut down to system state 1), a daemon process checks the error log. If the daemon finds outstanding bad block reports in a log, it sends a message to the system console. For example, in the scenario described earlier, the following message was sent:

```
Disk Error Daemon: Disk maj=17 min=0: 1 errors logged
```

The `hdelogger` daemon runs in system states 2, 3, and 4.

NOTE The bad block handling daemon does not run in system states 1, 5, or 6. System state 1 (also referred to as "s" or "S") is single-user state. System states 5 and 6 are used for returning to firmware and for rebooting the operating system, respectively.

Interactive Recovery from a Bad Block

Not all bad block errors can be handled automatically; recovery from some types of errors requires your assistance. Which recovery procedure is required depends on the system state of the computer when the error occurred.

Errors in System State 1 (Single-User State)

If errors happen while your system is in system state 1, the error reports stay queued until a system state is used in which the bad block handling daemon will also run. However, if you shut your system off or reboot it without going to another system state, the error reports in the queue are lost. When errors occur while in system state 1, only the messages from the logging mechanism and disk driver are sent to the system console.

If you get errors while in system state 1 and you are not ready to fix them (the mechanism for fixing them takes error reports from the queue as well as from the disk error logs), you can switch to another system state to force them to be logged. You will get a `successfully logged` message for each error that occurred. When all reports are logged, you can switch back to system state 1. When you do, a reminder message from the disk error daemon will be sent to the console.

System Panics and Firmware-Detected Errors

If there is a disk error that results in the loss of a critical operating-system path, such as the path to the swap space, the operating system panics. A system panic occurs after the reports from the logging mechanism and disk driver are sent to the console, but before the report is logged.

NOTE If an error is detected by the firmware, the error is reported to the console, but it is not logged. In these cases, YOU MUST WRITE DOWN THIS CONSOLE REPORT. When the system comes back up, use the `/usr/sbin/hdeadd` command and manually add this handwritten report to the disk error queue.

The following is an example of a bad block error message from the firmware.

```
id 0 CRC error at disk address 00010211
```

Because the arguments to the `hdeadd` and `hdefix` commands must be in decimal format, the hexadecimal disk address (`00010211`) must be converted to the decimal values of the cylinder, track, and sector. Figure 4-7 shows how the hexadecimal number is converted to decimal format, and how this decimal number translates to the address on the disk.

Figure 4-7: Sample Disk Address Conversion

	Physical Cylinder Number High		Physical Cylinder Number Low		Physical Head Number		Physical Sector Number	
Hex. Address	0	0	0	1	0	2	1	1
Binary	0000	0000	0000	0001	0000	0010	0001	0001
Decimal	1				2		17	
	Cylinder				Track		Sector	

In this example, the hexadecimal number `00010211` is translated to cylinder 1, track 2, and sector 17. These decimal numbers are then used as arguments to the `-B` option of the `hdeadd` and `hdefix` commands (that is, `-B 1 2 17`).

The following screen shows example messages that appear when a bad block causes a system panic. Assume that physical block number 463 is in your swap space. Although, unknown to you, this block has recently become bad, the operating system writes data into it. (These data cannot be read with the disk's current pattern of magnetic biases.) When the operating system tries to read this block, the disk driver determines that the block is unreadable, reports the condition, and fails the read. The swapper process runs next, discovers the read failure, and causes the panic. All process activity is precluded at this point, including the disk error daemon, and the following messages are sent to the console.

```
WARNING: unreadable CRC hard disk error: maj/min = 17/0

         block # = 463

WARNING:
hard disk: cannot access sector 13, head 0, cylinder 5, on drive 0

PANIC: swapseg: i/o error in swap
```

NOTE Because of the system panic, the system cannot record this error. Therefore you must write down or print out the numbers 17/0 and 463.

Unless you are already in system state 5, you must bring the system down to system state 1 with the `shutdown` command (`/sbin/shutdown -y -g0 -i1`) to minimize the chances of getting another swap-generated panic. Once you are in system state 1, enter the following command:

```
hdeadd -a -D 17 0 -b 463
```

This command reports the bad block to the operating system's logging mechanism. If this swap error occurred on Saturday, January 13, 1990, at 02:01:00 hours, the full report would be as follows:

```
# hdeadd -a -D 17 0 -b 463
hdeadd: logging the following error report:
        disk maj= 17 min=0
        blkaddr= 463, timestamp= Sat Jan 13 02:01:00 1990
        readtype= 1, severity= 2, badrtcnt= 0, bitwidth= 0
WARNING: unreadable CRC hard disk error: maj/min = 17/0

         block # = 463

Disk Error Daemon: successfully logged error for block 463 on disk maj= 17 min= 0
#
```

You can also use the `hdelogger -f` command to double check the error status and obtain the following report (assuming this is the only error in the log).

```
# hdelogger -f

Disk Error Log: Full Report for maj= 17 min= 0
        log created:  Mon Jan  1 12:13:14 1990
       last changed: Sat Jan 13 02:01:04 1990
        entry count: 1
        phys blkno      cnt     first occurrence        last occurrence
0:      463     1       Jul 13 02:01:00 1990    Jul 13 02:01:00 1990
TOTAL: 1 errors logged
#
```

If the firmware detected the error, it may not be possible to boot the system from your hard disk. However, if you have a floppy diskette device on your system and a bootable floppy containing the bad block utilities, you can boot from this diskette and try to repair the bad blocks as described in "Manual Recovery from a Bad Block" below. If you do not have a floppy diskette device on your system or if you do not have a bootable floppy containing the bad block utilities, contact your service representative.

The Special Case of a Bad Error Log Block

Though unlikely, the new bad block could be the block in which the disk error log resides. Obviously, if the system cannot access the log because this log resides in a bad block, errors cannot be recorded. An auxiliary mechanism of the `hdefix` command, however, allows you to add bad blocks to the defect map, as discussed next.

Manual Recovery from a Bad Block

To fix a bad block manually:

1. Completely shut down the machine to system state 1

   ```
   shutdown -y -g0 -i1
   ```

2. Unmount all file systems except root

   ```
   umountall -k
   ```

3. Force any queued reports from the disk error queue to the error log.

   ```
   /sbin/hdefix -a
   ```

   If an error is reported while you are in system state 1, you need not switch system states to get it processed.

The `hdefix -a` command updates the defect map appropriately (remember that bad surrogate blocks are replaced, not mapped), removes any reports for the block (there may be more than one) from the disk error log, and identifies the use of the block. If the bad block is in a file system, the file system is marked bad.

If any block (or surrogate of such a block) in the normally accessible portion of the disk has been processed, the `hdefix` command forces an immediate reboot that checks the integrity of the root file system while coming back up. You must manually check the integrity of any other file system you want to mount.

You can also specify which disk and block(s) are to be fixed by using additional arguments to the `hdefix` command. In the swap panic example above, you could have specified the bad block to be fixed by entering the following command from system state 1:

```
hdefix -a -D 17 0 -b 463
```

When you specify a block number, `hdefix` ignores the contents of the error log and the error queue. If there is a report in the log for this block, the report remains in the log after you have finished. As a result, when reviewing the log, you may incorrectly interpret it to mean this block is still bad. However, when you issue the command `hdefix -a`, the fix list is taken from the log and queue and the log is cleaned up automatically.

Dealing with Data Loss

Although the useful life of disk hardware is greatly extended with the bad block handling feature, once a bad block is logged, the data in the block are lost. You must be prepared to restore files or file systems from archives. When restoring data from archives, users may encounter unavoidable inconveniences in the form of lost files and lost work on existing files.

Under rare circumstances, a bad block may occur in the Volume Table of Contents (VTOC) block. In such a case, you may have to reformat the disk and restore the contents of the entire disk from an archive. Sometimes you may also need to restore the special code (the program) for booting the system (it is not in a file system).

Quick Reference to Diagnostics

- Accessing the diagnostic monitor:

  1. Enter system state 5 (firmware state) by executing

     ```
     shutdown -y -i5
     ```

     or using the express shutdown

     ```
     shutdown -y -i5 -g0
     ```

     The following messages will appear on the screen:

     ```
     SELF-CHECK
     FIRMWARE MODE
     ```

  2. Enter the firmware password.
  3. If you see either the > prompt or the < prompt, type

     ```
     boot
     ```

     Otherwise, go to Step 4.
  4. At the prompt, enter

     ```
     dgmon
     ```

  5. Display the Equipped Device Table by entering

     ```
     s
     ```

  6. Select the device to be tested.
  7. Run diagnostics as required using `dgn`.

- Displaying the Diagnostic Phase Table for a specific device:

  `l` *device_name*

  where *device_name* is the name of a specific device

- Displaying a help menu for `dgmon`:

  ```
  h
  ```

- Leaving the Diagnostic Monitor:

  ```
  q
  ```

- Running the diagnostic phases once on all devices in the Equipped Device Table:

  ```
  dgn
  ```

- Running the diagnostic phases once on all devices of one type:

  `dgn` *device_type*

  where *device_type* is the type of device on which diagnostic phases are to be run.

- Running diagnostic phases on a specific instance of a device:

  `dgn` *device_name #*

  where *device_name #* is the name and identification number of a specific device.

- Running a specific diagnostic phase:

  `dgn ph=`*n*[-*m*

  where *n* is the number of a specific phase and *n*–*m* are numbers defining a range of phases.

- Running multiple repetitions of diagnostic phases:

  `dgn rep=`*n*

  where *n* is the number of times you want the phases to run.

- Running diagnostic phases continuously:

  ```
  dgn soak
  ```

  where `soak` specifies that the phases are to be run continuously and the results are to be stored until testing is completed.

- Running diagnostic phases in the unconditional mode:

  ```
  dgn ucl
  ```

- Adding hand-written reports to the disk error queue:

  ```
  /usr/sbin/hdeadd
  ```

- Generating a hard disk error report:

  ```
  hdelogger -f
  ```

- Repairing a bad block that caused a system panic:

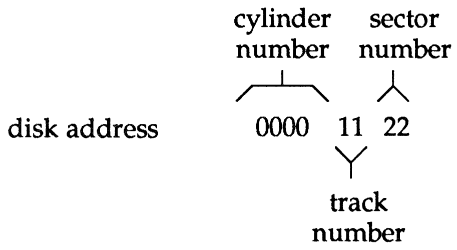
  1. Write down the appropriate information from the `PANIC` message displayed on the system console. (The disk address may be displayed in hexadecimal format or in major and minor device and bad block numbers.)
  2. Bring the system down to system state 1.

     ```
     shutdown -y -g0 -i1
     ```

  3. Log in as `root`.
  4. Kill all processes with open files and then unmount all file systems except root.

     ```
     /sbin/umountall -k
     ```

  5. If necessary, convert the hexadecimal disk address to decimal cylinder, track, and sector numbers.



  6. Report the bad block to the logging mechanism in either of two ways.

(a) Enter

```
/usr/sbin/hdeadd -a -D XX YY -b ZZ
```

where *XX* is the major device number, *YY* is the minor device number, and *ZZ* is the bad block number

(b) Enter

```
/usr/sbin/hdeadd -a -B aa bb cc
```

where *aa* is the decimal cylinder number, *bb* is the decimal track number, and *cc* is the decimal sector number.

7. Force all queued errors to the error log while the system is processing the disk.

   ```
   /sbin/hdefix -a
   ```

8. Test the integrity of each file system by entering

   ```
   /sbin/fsck -fD file_system
   ```

   where *file_system* is the full path of the file systems reported to have been unmounted with the `umountall -k` command (in Step 4), one file system at a time.

9. Remount all good file systems by entering

   ```
   /sbin/mount directory
   ```

   where *directory* is the mount point on which the file system will be mounted. The file system partition is described in `/etc/vfstab`.

10. Restore any lost data from system archives (refer to the ''Restore Service'' chapter).

11. Go to system state 2.

    ```
    init 2
    ```

5 File System Administration

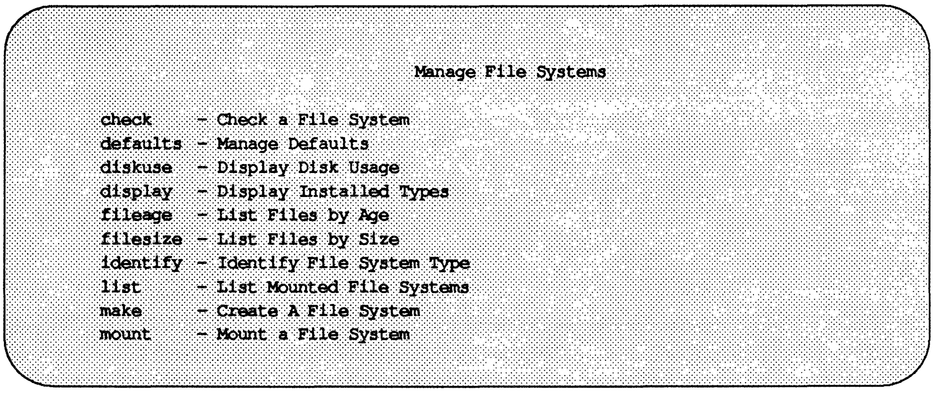
Prologue

This chapter provides the information required to administer disk file systems. Other file system types, such as /proc, are not discussed. (For information on /proc, see the manual page proc(4) in the *System Administrator's Reference Manual*. For information on remote file systems (e.g., `rfs` and `nfs`) see the *Network User's and Administrator's Guide*.)

You may approach file system administration in either of two ways: by using administrative menus or by issuing commands directly to the shell. Generally speaking, if you are new to system administration it is probably preferable to use the administrative menus, while if you are more experienced you may appreciate the greater speed and flexibility that may be attained by entering commands at the shell level.

To perform administrative tasks using the menus provided by the system, you must type `sysadm file_systems`. When you do so, you will reach the main menu for file system administration, which is shown below.

Figure 5-1: Main Menu for File System Administration



```
                              Manage File Systems

     check     - Check a File System
     defaults  - Manage Defaults
     diskuse   - Display Disk Usage
     display   - Display Installed Types
     fileage   - List Files by Age
     filesize  - List Files by Size
     identify  - Identify File System Type
     list      - List Mounted File Systems
     make      - Create A File System
     mount     - Mount a File System
```

The menus are intended to be self-explanatory and will not be further described in this chapter.

If you prefer not to use the menus, you can perform administrative tasks by issuing commands directly to the shell. The following table shows the shell commands that are functionally equivalent to the selections on the main menu for file system administration.

Figure 5-2: Menus and Shell Commands for Performing Administrative Tasks

Task Description	Menu Item	Shell Command
Check a file system	`check`	`fsck`(1M)
Manage `vfstab` defaults	`defaults`	any editor
Display disk usage	`diskuse`	`df`(1M)
Display installed types	`display`	`crash`(1M)
List files by age	`fileage`	`ls -t`
List files by size	`filesize`	du(1M)
Identify a file system type	`identify`	`fstyp`(1M)
List file systems	`list`	`mount`(1M)
Create a file system	`make`	`mkfs`(1M)
Mount a file system	`mount`	`mount`(1M)
Unmount a file system	`umount`	`umount`(1M)

Many of the tasks listed in this table are described in detail later in this chapter. Complete information about each shell command is available in the *System Administrator's Reference Manual*.

Introduction

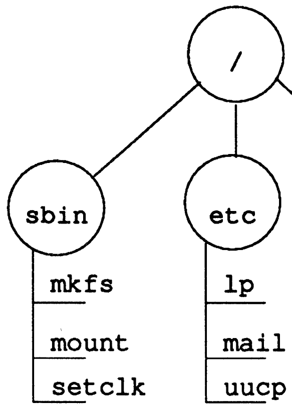
UNIX System V Release 4.0 supports a variety of file system types (FSTypes) of varying characteristics. Because of the great range of possible FSTypes, it is impossible to provide administrative information that would apply to every conceivable type. The material presented here describes the administration of the FSTypes `s5`, `ufs`, and `bfs` (the boot FSType).

We begin by describing the `s5`, `ufs`, and `bfs` FSTypes. We then discuss the relationship between a file system and a storage device, the methods of administering and maintaining a file system, and how to check a file system for consistency. In each section separate subsections describe the methods applicable to each of the FSTypes under discussion.

How a File System Is Organized

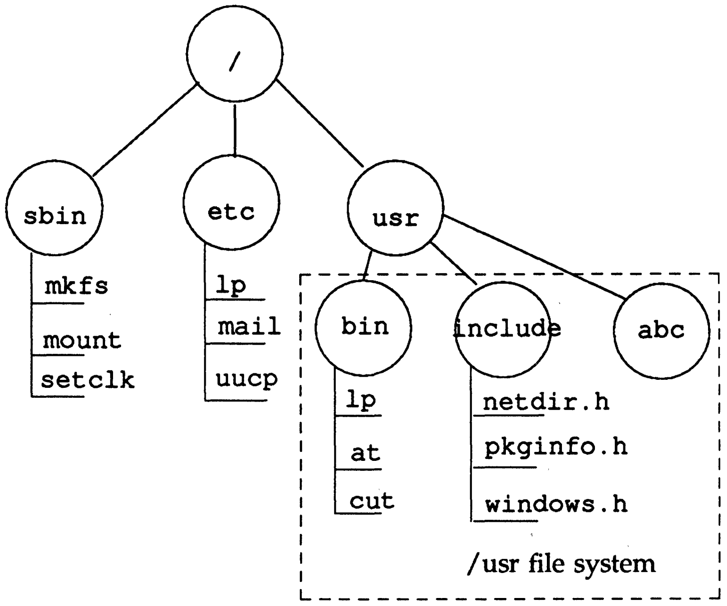
A primary function of the UNIX operating system is to support file systems. In the UNIX system a file is a string of bytes with no other structure implied. Files are attached to a hierarchy of directories. A directory is merely another type of file that the user is permitted to use, but not to write; only the operating system can write directories. The combination of directories and files make up a file system. Figure 5-3 shows the relationship between directories and files in a UNIX file system. The circles represent directories.

Figure 5-3: A UNIX File System



The starting point of any UNIX file system is a directory that serves as the root of that file system. Somewhat confusingly, in the UNIX operating system there is always one file system that has the name `root`. Traditionally, the root directory of the `root` file system is represented by a single slash (/). The file system diagrammed in Figure 5-3 is a `root` file system. If we mount another file system onto the `root` file system at a directory called `/usr`, the result can be illustrated by the diagram in Figure 5-4.

Figure 5-4: Adding the `/usr` File System



A directory such as `/usr` that is used to form the connection between the `root` file system and another mountable file system is sometimes called a "leaf" or "mount point." Regardless of the term used, such a directory is the root of the file system that descends from it. The name of that file system is the name of the directory. In our example the name of the file system is `/usr`.

The diagrams in Figures 5-3 and 5-4 may be convenient representations of the file and directory structure of file systems, but they are not a particularly accurate or helpful way of illustrating how the UNIX operating system views a file system. The sections that follow describe FSTypes as they appear to the operating system.

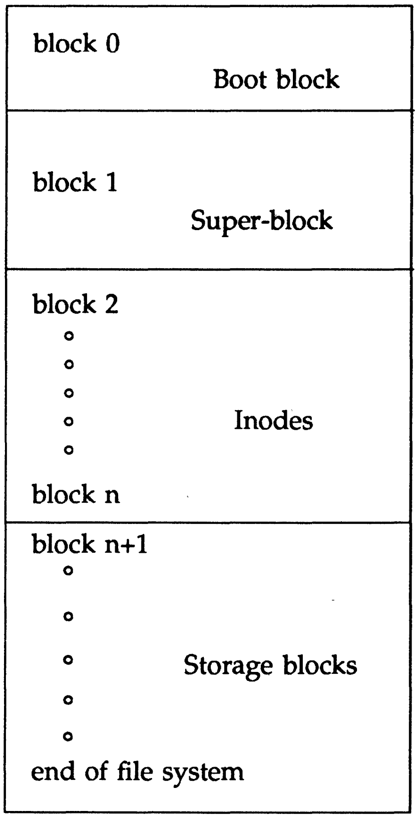
The s5 File System Type

The operating system views an s5 file system as an arrangement of addressable blocks of disk space that belong to one of four categories:

- block 0 (the boot block)
- block 1 (the super-block)
- a variable number of blocks comprising the i-list
- a variable number of storage blocks, most containing data, some containing the free list and indirect addresses

This layout is illustrated in Figure 5-5.

Figure 5-5: The UNIX View of an s5 File System



The s5 Boot Block

Although considered to be part of the file system, the boot block is not actually used by it. It is reserved for storing procedures used in booting the system. However, not all file systems are involved in booting. When a file system is not to be used for booting, the boot block is left unused.

The s5 Super-Block

Much of the information about the file system is stored in the super-block, including such things as:

- file system size and status
    - label (file system name)
    - size in logical blocks
    - read-only flag
    - super-block modified flag
    - date and time of last update
- inodes
    - total number of inodes allocated
    - number of free inodes
    - array of 100 free inode numbers
    - an index into the array of free inode numbers
- storage blocks
    - total number of free blocks
    - array of 50 free block numbers
    - an index into the array of free block numbers

Note that the super-block does not maintain complete lists of free inodes and free blocks, but only enough to meet current demands as the file system is used. At almost any time, unless the file system is close to running out of inodes and storage blocks, there are sure to be more free inodes and free blocks than are listed in the super-block.

s5 I-Nodes

The term "inode" (or inode) stands for information node. A list of inodes is called an i-list and the position of an inode in an i-list is called an i-number.

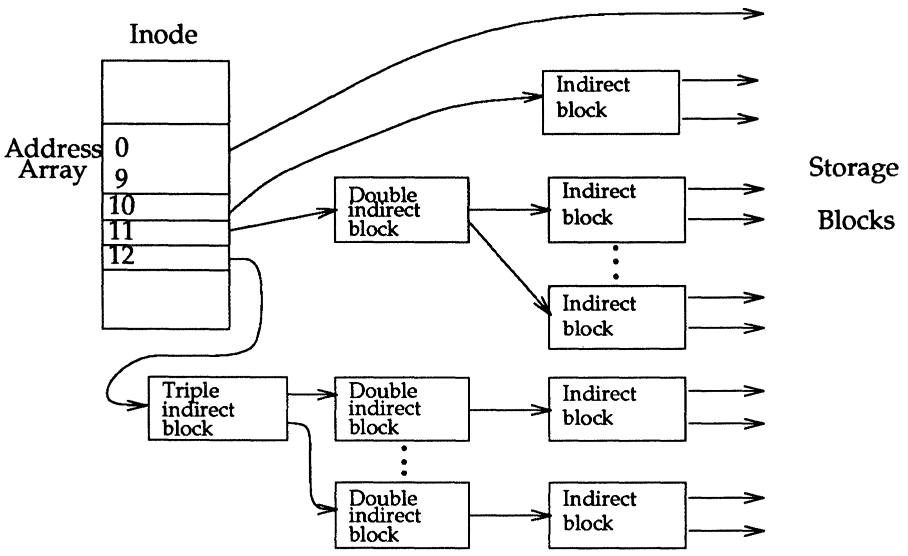
An inode contains all the information about a file except its name, which is kept in a directory. Inodes are 64 bytes long, so there are 8 of them in a physical block. The length of an i-list is not fixed; it depends on the number of inodes specified when the file system is created. Specifically, an s5 inode contains:

- the type and mode of the file – the type can be regular, directory, block, character, symbolic link, or FIFO, also known as named pipe; the mode is the set of read-write-execute permissions
- the number of hard links to the file
- the user-id of the owner of the file
- the group-id to which the file belongs
- the number of bytes in the file
- an array of 13 disk block addresses
- the date and time the file was last accessed
- the date and time the file was last modified
- the date and time the file was created

The array of 13 disk block addresses is the heart of the inode. The first 10 are direct addresses; that is, they point directly to the first 10 logical storage blocks of the contents of the file. If the file is larger than 10 logical blocks, the 11th address points to an indirect block, which contains direct addresses instead of file contents; the 12th address points to a double indirect block, which contains addresses of indirect blocks. Finally, the 13th address in the array is the address of a triple indirect block, which contains addresses of double indirect blocks.

Figure 5-6 illustrates this chaining of address blocks stemming from the inode.

Figure 5-6: The File System Address Chain for s5



The following table shows the number of bytes addressable by the different levels of indirection in the inode address array for s5 file systems. These numbers are calculated using the logical block size of the file system and the number of bytes (4) used to hold an address.

	Maximum number of bytes addressable by			
Logical Block Size	Direct Blocks	Single Indirect Blocks	Double Indirect Blocks	Triple Indirect Blocks
512 bytes	5120	64K	8M	1G
1024 bytes	10240	256K	64M	16G
2048 bytes	20480	1M	512M	256G

The table shows the number of bytes addressable using the level of indirection in the column header plus all lower levels of addressing. For example, the table values for single indirect blocks also include bytes addressable by direct blocks; and the table values for triple indirect blocks include bytes addressable by direct blocks and single and double indirect blocks.

The theoretical maximum size of an s5 system file is the same as the size of a file addressable with triple indirection (shown in the last column of the table). In practice, however, file size is limited by the size field in the inode. This is a 32-bit field, so file sizes are limited to 4 gigabytes.

s5 Storage Blocks

The rest of the space allocated to the file system is occupied by storage blocks, also called data blocks. For a regular file, the storage blocks contain the contents of the file. For a directory, the storage blocks contain 16-byte entries. Each entry represents a file or subdirectory that is a member of the directory. An entry consists of 2 bytes for the i-number and 14 bytes for the filename of the member file or subdirectory.

s5 Free Blocks

Blocks not currently being used as inodes, as indirect address blocks, or as storage blocks are chained together in a linked list of free blocks. Each block in the list carries the address of the next block in the chain.

The `ufs` File System Type

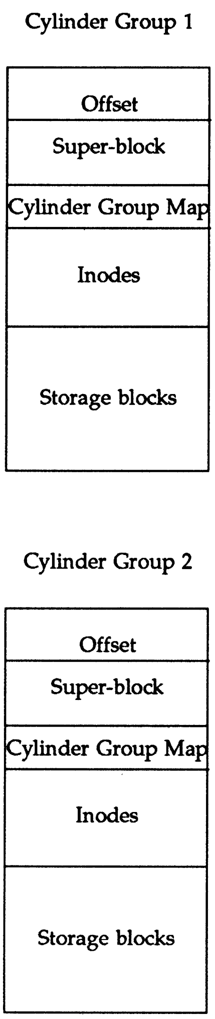
The `ufs` FSType is considerably more complex in its design than the `s5` FSType. In addition to the four categories of addressable blocks found in `s5`, there are several additional information management disk areas. There is also a radically different method of allocating and managing these blocks. Of primary interest is the fact that multiple super-blocks are made during the `mkfs` procedure. One of the replicas is stored in each cylinder group, offset by a certain amount. For multiple platter disk drives, the offsets are calculated so that a super-block appears on each platter of the drive. So if the first platter is lost, an alternate super-block can be retrieved. For platters other than the top one in a pack, the leading blocks created by the offsets are reclaimed for data storage.

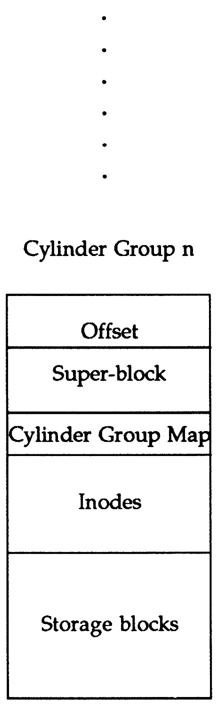
Kept with the super-block is a summary information block. This block is not replicated, but is grouped together with the first super-block, normally in cylinder group 0. This summary block is used to record changes that take place as the file system is used, and lists the number of inodes, directories, fragments, and blocks within the file system.

Another feature found in `ufs` is the "cylinder group map." This is a block of data found in each cylinder group that records the block usage within the cylinder. This information is kept directly following the super-block copy for that cylinder group.

To give an idea of the appearance of a typical `ufs` file system, the following diagram shows a series of cylinder groups in a generic `ufs` file system:

Figure 5-7: The UNIX View of a `ufs` File System





The `ufs` Boot Block

The boot block appears only in the first cylinder group (cylinder group 0) and is the first 8K in a partition. It is reserved for storing the procedures used in booting the system. If a file system is not to be used for booting, the boot block is left blank.

The `ufs` Super-Block

Much of the information about the file system is stored in the super-block. A few of the more important things it contains are:

- the size and status of the file system
- the label (file system name)
- the size of the file system in logical blocks
- the date and time of the last update
- the cylinder group size
- the number of data blocks in a cylinder group
- the summary data block

`ufs` I-Nodes

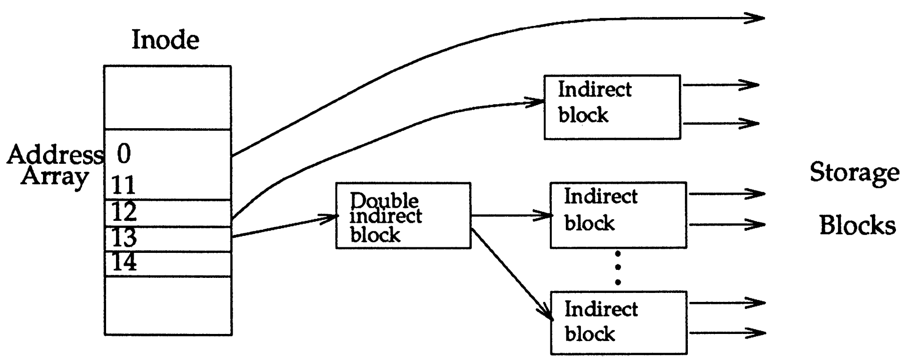
The inode information is kept in the cylinder information block. An inode contains all the information about a file except its name, which is kept in a directory. An inode is 128 bytes long. One inode is created for every 2K of storage available in the file system. This parameter can be changed when `mkfs` is used to create the file system, but it is fixed thereafter. A `ufs` inode contains:

- the type and mode of the file – the type can be regular, directory, block, character, symbolic link, or FIFO, also known as named pipe; the mode is the set of read-write-execute permissions
- the number of hard links to the file
- the user-id of the owner of the file
- the group-id to which the file belongs
- the number of bytes in the file
- an array of 15 disk block addresses
- the date and time the file was last accessed

- the date and time the file was last modified
- the date and time the file was created

The array of 15 disk addresses is the heart of the inode. The first 12 are direct addresses; that is, they point directly to the first 12 logical storage blocks of the contents of the file. If the file is larger than 12 logical blocks, the 13th address points to an indirect block, which contains direct addresses instead of file contents; the 14th address points to a double indirect block, which contains addresses of indirect blocks. The 15th address is unused. Figure 5-8 illustrates this chaining of address blocks stemming from the inode.

Figure 5-8: The File System Address Chain in a `ufs` File System



The following table shows the number of bytes addressable by the different levels of indirection in the inode address array for `ufs` file systems. These numbers are calculated using the logical block size of the file system and the number of bytes used to hold an address.

	Max number of bytes addressable by		
Logical Block Size	Direct Blocks	Single Indirect Blocks	Double Indirect Blocks
4096 bytes	48K	4M	4G
8192 bytes	96K	16M	32G

The table shows the number of bytes addressable using the level of indirection in the column header plus all lower levels of addressing. For example, the table values for single indirect blocks also include bytes addressable by direct blocks; and the table values for indirect blocks include bytes addressable by direct blocks and single indirect blocks.

The theoretical maximum size of a `ufs` file system is the same as the size of a file addressable with double indirection. In practice, however, file size is limited by the size field in the inode. This is a signed 32-bit field, so file sizes are limited to 2 gigabytes. Because of the large size of `ufs` logical blocks, double indirect blocks rarely appear in `ufs` file systems. The result is that data retrieval in large files is much quicker than it would otherwise be.

`ufs` Storage Blocks

The rest of the space allocated to the file system is occupied by storage blocks, also called data blocks. The size of these storage blocks is determined at the time a file system is created, and can be either 4096 or 8192 bytes. Because of these large block sizes, and the potential for waste with small files, `ufs` also has a subdivision of a block called a fragment. When a file system is created the fragment size may be set to 512, 1024, 2048, or 4096 bytes. Fragments of 1024 bytes are the most commonly employed. For a regular file, the storage blocks contain the contents of the file. For a directory, the storage blocks contain entries that give the inode number and the filename. `ufs` filenames may be up to 255 bytes long. Each entry represents a file or subdirectory that is a member of the directory.

ufs Free Blocks

Blocks not currently being used as inodes, as indirect address blocks, or as storage blocks are marked as free in the block map kept in the cylinder group summary information block. This block also keeps track of fragments in order to prevent fragmentation from degrading disk performance excessively.

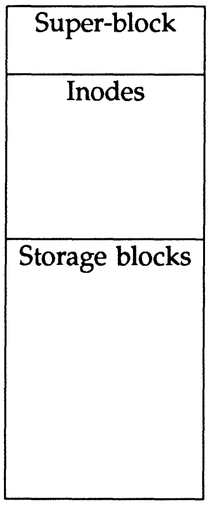
The `bfs` File System Type

The `bfs` FSType is a special-purpose file system. It contains all the stand-alone programs (e.g. `unix`) and all the text files necessary for the boot procedures. See "The `stand` and `boot` Partitions" in Chapter 6, "Machine Management" for more information.

The object of the `bfs` FSType is to allow quick and simple booting. It is for this reason that `bfs` was designed as a contiguous flat file system. The only directory `bfs` supports is the `root` directory. Users may only create regular files; no directories or special files can be created in the `bfs` file system.

A `bfs` file system consists of three parts: the disk super-block, the inodes, and the storage or data blocks. The layout is illustrated in the following figure:

Figure 5-9: The UNIX View of a `bfs` File System



The bfs Super-Block

The following information is stored in the super-block:

- the magic number
- the size of the file system
    - the offset to the start of file system data (in bytes)
    - the offset to the end of file system data (in bytes)
- the sanity words

  There are four words used to promote sanity during compaction. They are used by the `fsck` command to recover if there has been a system crash at any time during the process of compaction. See "Compaction" in this chapter for more information about compaction.

bfs I-Nodes

The inode contains all the information about a file except its name. File names are kept in the `root` directory, the only directory in the `bfs` file system. An inode is 64 bytes long. The number of inodes is defined when `mkfs` is used to create the file system. An inode contains:

- the inode number

  By convention this field is set to zero to indicate that the inode is available.
- the first data block
- the last data block
- the disk offset to the end-of-file (in bytes)
- the file attributes
- the type and mode of the file

- the user ID of the owner of the file
- the group ID to which the file belongs
- the number of hard links to the file
- the date and time the file was last accessed
- the date and time the file was last modified
- the date and time the file was created

bfs Storage Blocks

The remainder of the space allocated to the file system is taken up by storage blocks, also called data blocks. The size of the storage blocks is 512 bytes. The storage blocks are used to store the `root` directory and the regular files. For a regular file, the storage blocks contain the contents of the file. For the `root` directory, the storage blocks contain 16-byte entries. Each entry represents a file and consists of 2 bytes for the i-number and 14 bytes for the file name.

Managing Data Blocks

The data or storage blocks for a file are allocated contiguously. The data block after the last data block used in the file system is considered the next data block available to store a file. When a file is deleted, its data blocks are released; for the file system to reuse them, one of the following must be true:

- the file deleted must be the last file stored in the file system, or
- the system must detect the need for compaction and perform it.

Compaction

Compaction is a way of recovering data blocks by shifting files until the gaps left behind by deleted files are eliminated. This operation can be very expensive, but it is necessary because of the method used by `bfs` to store and delete files.

The system recognizes the need for compaction and performs it when:

- the system has reached the end of the file system and there are still free blocks available, or
- the system deletes a very large file and the file after it on disk is small and is the last file in the file system. (Small files are files of no more than 10 blocks; large files are files of 500 or more blocks.)

The Relationship Between the File System and the Storage Device

In the UNIX system, file systems are kept on random-access disk devices. (A file system can be put on tape, but that is normally done only to create a backup copy in case the file system must be restored.) Before you can install a file system on a storage device, you must format the device. The material in this part of the chapter is a summary of material described in more detail in Chapter 15, ''Storage Device Management''.

Formatting the Storage Device

Before a disk can be used by the UNIX system it must be formatted into addressable sectors. A disk sector is a 512-byte portion of the storage medium that can be addressed by the disk controller. The number of sectors is a function of the size and number of surfaces of the disk device. Sectors are numbered from zero on up.

NOTE Hard disk units on the 3B2 Computer are formatted when installed. The only time they might have to be reformatted would be after a catastrophic hardware failure. We recommend that you contact your service representative if such an event occurs.

Diskettes are made to be usable in more than one machine. Manufacturers produce diskettes unformatted, leaving it to customers to format them for the particular machine on which they are to be used. The command to use to format a diskette is `fmtflop`(1M). A related command, `format`(1M) is used to format a SCSI hard disk.

Partitions

A partition consists of one or more sectors. Up to 16 partitions can be defined on a hard disk, up to 8 on a floppy diskette. On a hard disk, the `fmthard`(1M) command is used to associate the starting points of partitions with sector numbers. (See Appendix A for information on partitions on a floppy disk.) `prtvtoc` may be used to see the partitions assigned. `fmthard` gives the number of sectors allocated to a partition and a hex code tag that tells how the partition is to be used. Partition tags 0-8 are reserved. The list below shows how the tags may be used under `s5`.

Name	Number
UNASSIGNED	0
BOOT	1 or 0
ROOT	2
SWAP	3
USR	4
BACKUP	5 or 0
STAND	6
VAR	7
HOME	8

The following figure is an example of the partitions for a 72-megabyte disk drive on a 3B2 Computer that has been configured to be the root device of a system.

Figure 5-10: Disk Partitions for a 72-Megabyte Drive

72-Megabyte Hard Disk Drive (blocks per cylinder = 162) (rotational gap = 10)				
Disk Partition	Use	Sector Start	Size*	I-Nodes
`c1d0s0`	`root`	14256	17010	2112
`c1d0s1`	`swap`	100	14156	–
`c1d0s2`	`usr`	36774	96066	12000
`c1d0s3`	`stand`	31266	5508	55
`c1d0s4`	unassigned			
`c1d0s5`	unassigned			
`c1d0s6`	entire disk	0	149526	–
`c1d0s7`	`boot`	0	100	–
`c1d0s8`	`var`	132840	10044	1248
`c1d0s9`	home	142884	6642	800
`c1d0sa`	unassigned			
`c1d0sb`	unassigned			
`c1d0sc`	unassigned			
`c1d0sd`	unassigned			
`c1d0se`	unassigned			
`c1d0sf`	unassigned			

* Size in 512-byte blocks.

The table shows five file systems defined on this drive: `root`, `/usr`, `/stand`, `/var`, and `/home`. Space has also been set aside for the boot file and for swap space. Tables showing the default partitioning for all supported devices are in Appendix A.

Size Limitations

The maximum number of blocks that can be allocated to a file system is close to the total number of sectors on the disk device. This maximum may be reduced by space set aside for swapping or paging. Also, recall that under `ufs` a 2 gigabyte upper limit is imposed by the size field in the inode.

The maximum number of inodes that can be specified for `s5` is 65,500. The `ufs` FSType does not have a rigid upper limit. Rather, roughly one inode is created automatically for each 2K of data space, although this default can be overridden at the time the file system is created.

The size of a block on a disk is 512 bytes, the same as a disk sector. However, internal file I/O works with logical blocks rather than with 512-byte physical blocks. The size of a logical block is set with the `mkfs`(1M) command. `s5` uses logical block sizes of 512, 1024, and 2048 bytes; the default is 1024 byte blocks. `ufs`, on the other hand, uses logical block sizes of 4096 and 8192 bytes; the default is 8192 byte blocks.

Because of the large block size used by `ufs`, small files could waste a lot of space. To deal with this, `ufs` has a subdivision of a block called a fragment. When a `ufs` file system is created using `mkfs`, the fragment size may be set to 512, 1024, 2048, or 4096 bytes. When a block must be fragmented, the remaining fragments are made available to other files to use for storage. The information on which fragments are in use and which are available is kept in the cylinder group summary information block.

Administering a File System

The Generic Administrative Commands

The Virtual File System architecture allows multiple file system types (FSTypes) to coexist in the UNIX kernel. Each FSType has certain characteristic features that it does not share with any other FSTypes. However, the file system administrative commands provide a common interface that allows the administrator to maintain file systems of differing types.

The following commands for file system administration are unified commands and can be used on multiple FSTypes:

- `dcopy`(1M)
- `df`(1M)
- `ff`(1M)
- `fsck`(1M)
- `fsdb`(1M)
- `fstyp`(1M)
- `labelit`(1M)
- `mkfs`(1M)
- `mount`(1M)
- `mountall`(1M)
- `ncheck`(1M)
- `umount`(1M)
- `umountall`(1M)
- `volcopy`(1M)

Most of these commands may be invoked as follows:

command `[-F` *FSType*`]` `[-V]` `[`*current_options*`]` `[-o` *specific_options*`]` *operands*

The `-F` is used to specify the FSType on which the command must act. The FSType must be specified on the command line or must be determinable from `/etc/vfstab` by matching an entry in that file with one of the *operands* specified. (See the section below for information on the `vfstab` file.)

The `-V` option causes the command to echo the completed command line. The echoed line will include additional information derived from the `vfstab` file. This option can be used to verify and validate the command line. It does not cause the command to execute.

current_options are options supported by the `s5`-specific module of the *command.*

The `-o` option is used to specify FSType-specific options. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the FSType-specific module of the command.

operands are FSType-specific; the FSType- specific manual page of the command should be consulted for a detailed description.

The `vfstab` File

Since the generic commands work on multiple FSTypes - for example, `mount` can mount an `s5` or a `ufs` file system among other types - they require FSType-specific information which may be provided explicitly on the command line or implicitly through the file system table `/etc/vfstab`.

The file system table contains a list of default parameters for each file system. It is an ASCII file that should be maintained by the system administrator. Each record contains space separated information about a file system in the format:

special fsckdev mountp fstype fsckpass automnt mntopts

The meaning of each field is as follows:

- *special*: the block special device for local devices or the resource name for remote file systems (e.g. `rfs` and `nfs`). (For more information on remote file systems, see the *Network Applications Guide*).
- *fsckdev*: the character special device that corresponds to the *special.* The block special device is used if the character special device is not available. Use a "–" where there is no applicable device.
- *mountp*: the default mount directory (mount point)
- *fstype*: the type of the file system on the special device

- *fsckpass*: the pass number to be used by `ff`, `fsck`, and `ncheck` to decide whether to check the file system automatically. Use "–" to inhibit automatic checking of the file system.
- *automnt*: `yes` or `no` for whether the file system should be automatically mounted by `mountall` when the system is booted.
- *mntopts*: a list of comma-separated options that will be used in mounting the file system. Use "–" to indicate no options. See `mount`(1M) for a list of the available options.

Lines beginning with the # character are comments.

Listing Installed File System Types

Use the `crash`(1M) command to display a list of FSTypes installed in the kernel. The following will produce such a list:

```
crash <<!
vfssw
!
```

In addition to the familiar FSTypes, `crash` will also list certain internal FSTypes, such as `specfs`, that have no user interface.

Identifying the Type of an Unmounted File System

Most commands that are used in file system administration require that the FSType of a file system be provided on the command line or in the file system table. Most of these commands also attempt to distinguish the type of a file system by themselves, so if the administrator provides the wrong type the command may fail. However, it is important to specify the correct type because file systems may be damaged if a command fails to detect an administrator's error and an operation applicable only to one type of file system is applied to another.

Sometimes, the administrator will have to try to determine the type of an unmounted file system type either because the `vfstab` file contains outdated information or because it contains no information at all. The command `fstyp`(1M) uses heuristics to determine the type of an unmounted file system. `fstyp` determines and displays the file system type on `stdout`. If it cannot determine the type it echoes *unknown_fstyp(no matches)* on `stderr`.

Creating a File System

Using `mkfs`

Once a disk is formatted the next step is to define the file system. The `mkfs`(1M) command is used for this purpose. The generic format of the `mkfs(1M)` command follows:

```
mkfs [F FSType] [-V] [-m] [current_options] [-o specific_options] \
    special [operands]
```

(The above command line is shown on two lines for readability.)

`mkfs` constructs a file system by writing on the *special* file, which must be the first argument. The file system is created based on the *FSType* specified using the `-F` option, the *specific_options,* and *operands* specified on the command line.

The `-F` is used to specify the *FSType* on which the command must act. The *FSType* must be specified on the command line or must be determinable from `/etc/vfstab` by matching an entry in that file with one of the *operands* specified. (See the section below for information on the `vfstab` file.)

The `-V` option causes the command to echo the completed command line. The echoed line will include additional information derived from `/etc/vfstab`. This option can be used to verify and validate the command line. It does not cause the command to execute.

The `-m` is used to return the command line which was used to create the file system. The file system must already exist and this option provides a means of determining the attributes used in constructing the file system. Note that file systems cannot be constructed for all *FSTypes*. Care must be taken to specify valid *FSTypes*.

current_options are options supported by the s5-specific module of the *command.*

The -o option is used to specify *FSType*-specific options if any. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

operands are FSType-specific; the *FSType*-specific manual page of the command should be consulted for a detailed description.

Choosing Logical Block Size

Logical block size is the size of the blocks that the UNIX kernel uses to read or write files. The logical block size is usually different from the physical block size, which is the size of the smallest block that the disk controller can read or write, usually 512 bytes.

The mkfs(1M) command allows the administrator to specify the logical block size of the file system. By default, the logical block size is 1024 bytes (1K) for s5 file systems and 8192 bytes (8K) for ufs file systems. The root and usr file systems are delivered as s5 2048-byte (2K) file systems. Besides 1K and 2K file systems, the s5 file system also supports 512-byte file systems while the ufs file system supports 4096-byte (4K) as well as 8K systems.

To choose a reasonable logical block size for your system, you must consider both the performance desired and the available space. For information on disk performance, see the section "Improving Disk Utilization" in the "Performance Management" chapter. For most ufs systems, an 8K file system with a 1K fragment size gives the best performance, while for most s5 systems, a 1K file system provides the best performance: both offer a good balance between disk performance and use of space in primary memory and on disk. For special applications running under s5 (such as s5 file servers) that use large executable files or large data files, a 2K file system may be a better choice. See the "Performance Management" chapter for more information.

Using mkfs to Create an s5 File System

When used to make an s5 file system, the mkfs command builds a file system with a root directory and a lost+found directory. It is usually invoked in one of the following ways:

```
mkfs [-F s5] [-b blocksize] special blocks[:inodes] [gap blocks/cyl]
mkfs [-F s5] [-b blocksize] special proto [gap blocks/cyl]
```

As discussed earlier, the file system type (`s5`) need only be specified on the command line if no entry has been set up for *special* in the `vfstab`(4) file by the administrator. See the section "The `vfstab` File".

Notice that neither form of invocation names the file system that is to be created (for this function see `labelit`(1M)); instead, both forms identify the file by the filename of the special device file on which it will reside. The special device file, traditionally located in the directory `/dev`, is associated with the identifying controller and unit numbers (major and minor, respectively) for the physical device.

In the first form of invocation the only other information that must be furnished on the `mkfs` command line is the number of 512-byte blocks the file system is to occupy. The second form lets you include that information in a prototype file that can also define a directory and file structure for the new file system, and it even allows for reading in the contents of files from an existing file system.

Both forms of invocation let you specify information about the interrecord gap and the blocks per cylinder. If this information is not given on the command line, default values are used. Figure 5-11 shows the recommended values for use with the `mkfs` command for devices using the `s5` file system type. (The recommended values are different from the defaults used by the command, which are: gap 10, size 162.) The recommendations depend on the logical block size of the file system; the `-b` option to `mkfs` lets you specify this. By default, the file system has a logical block size of 1024 bytes. With the `-b` option, you can specify a logical block size of 512 bytes, 1024 bytes, or 2048 bytes. In the first form of invocation, even though the number of blocks in the file is required, the number of inodes may be omitted. If the number of inodes is omitted, the command uses a default value of one inode for every four logical storage blocks, rounding down to a modulo 16 value if necessary to fill the final inode block.

Figure 5-11: Interrecord Gap and Blocks/Cylinder Recommendations for s5

Device	Gap Size 512-byte FS	Gap Size 1K FS	Gap Size 2K FS	Blks/Cyl	
10M Hard Disk	8	10	12	72	
30M Hard Disk	8	10	12	90	
72M Hard Disk	8	10	12	162	(CDC Wren II)
72aM Hard Disk	8	10	12	144	(Micropolis)
72bM Hard Disk	8	10	12	162	(Priam)
72cM Hard Disk	8	10	12	198	(Fujitsu)
Floppy Disk	1	1	1	18	

If you use the first form of invocation, the file system is created with a `root` directory and a `lost+found` directory. If you use a prototype file, as noted above, it may include information that allows the command to build and initialize a directory and file structure for the file system. The format of a prototype file is described on the `mkfs`(1M) manual page of the *System Administrator's Reference Manual.*

Summary: Creating and Converting s5 File Systems

Here is a summary of the steps in creating a new file system or converting an old one to a new logical block size:

1. If the new file system is to be created on a disk partition that contains an old file system, back up the old file system. For information, see the chapters on backup and restore in this guide; to back up systems with one or more hard disks use `cpio`(1).

2. If the new file system is to be created from an old file system, run the `labelit` command, which reports the mounted file system name and the physical volume name of the old file system (see `volcopy`(1M).) These labels are destroyed when you make the new file system, so you must restore them.

3. If the new file system is to be created from an old file system and the new file system will have a larger logical block size, then, because of fragmentation, the new file system will allocate more disk blocks for data storage

than the old. Use the `fsba`(1M) command to find out the space requirements of the old file system with the new block size.

Use the information you get from the `fsba` command to make sure that the disk partition to be used for the new file system is large enough. Use the `prtvtoc`(1M) command to find out the size of your current disk partitions. If the new file system requires a disk repartition, see "Formatting Floppy Diskettes, Hard Disks, and Tapes" in Chapter 15.

4. Use the `mkfs`(1M) command with the `-b` option to make the new file system with the appropriate logical block size.
5. Run the `labelit` command to restore the file system and volume names.
6. Populate the new file system—for example do a restore from a file system backup, or, if your system has two hard disks, do a `cpio` from a mounted file system. (The `volcopy`(1M) and `dd`(1M) commands copy a file system image; they cannot convert logical block size.)

Using `mkfs` to Create a `ufs` File System

When used to make a `ufs` file system, the `mkfs` command builds a file system with a `root` directory and a `lost+found` directory. The number of inodes is calculated as a function of the file system size.

The syntax for the `mkfs` command when it is used to create a `ufs` file system is the following:

```
mkfs -F ufs [-o specific_options] special size
```

The *specific_options* are a comma-separated list that allow you to control the parameters of the file system. The more important ones are as follows (for a complete list, see the `ufs`-specific `mkfs`(1M) manual page):

- `nsect` - The number of sectors per track on the disk. The default is 18.
- `ntrack` - The number of tracks per cylinder on the disk. The default is 9.
- `bsize` - The primary block size for files on the file system. It must be a power of two, currently selected from 4096 or 8192 (the default).

- `fragsize` - The smallest amount of disk space that will be allocated to a file. It must be a power of two, currently selected from the range 512 to 8192. The default is 1024.
- `cgsize` - The number of disk cylinders per cylinder group. This number must be in the range 1 to 32. The default is 16.
- `free` - The minimum percentage of free disk space allowed. Once the file system capacity reaches this threshold, you must be a privileged user to allocate disk blocks. The default is 10.

A sample invocation follows:

```
mkfs -F ufs -o bsize=4096,nsect=18,ntrack=9 \
/dev/rdsk/c0d0s2 35340
```

(The above command line is shown on two lines for readability.)

Summary: Creating and Converting `ufs` File Systems

Here is a summary of the steps required to create a new file system or convert an old one to a new logical block size:

1. If the new file system is to be created on a disk partition that contains an old file system, back up the old file system. For information, see the chapters on backup and restore in this guide; to back up systems with one or more hard disks use `cpio`(1).
2. If the new file system is to be created from an old file system, run the `labelit` command, which reports the mounted file system name and the physical volume name of the old file system (see `volcopy`(1M).) These labels are destroyed when you make the new file system, so you must restore them.
3. Use the `mkfs`(1M) command to make the new file system with the appropriate logical block size. The `mkfs`(1M) command is described in the section "Using `mkfs` to Create a `ufs` File System" in this guide.
4. Run the `labelit` command to restore the file system and volume names.
5. Populate the new file system—for example do a restore from a file system backup, or, if your system has two hard disks, do a `cpio`(1M) from a mounted file system. (The `volcopy`(1M) and `dd`(1M) commands copy a file system image; they cannot convert logical block size.)

Using `mkfs` to Create a `bfs` File System

When used to make a `bfs` file system, the `mkfs` command builds a file system with a `root` directory.

The syntax for the `mkfs` command when making a `bfs` file system is as follows:

`mkfs` [`-F bfs`] *special blocks* [*inodes*]

If the number of inodes is not specified on the command line, the default number of inodes is calculated as a function of the file system size.

Although any disk can have multiple boot file systems defined on it, you will not normally want more than one boot file system on one disk.

The following procedure shows how to define a new boot file system and assumes that the disk you are using is already bootable. See the section "Making New Bootable Disks" in the "Machine Management" chapter for instructions on making new bootable disks.

Defining a New Boot File System on a Bootable Disk

1. Use the `prtvtoc`(1M) command to identify the type and size of the current disk partitions on the disk. If your new `bfs` file system requires a disk repartition, see "Making New Bootable Disks" in the "Machine Management" chapter for information on partitioning a bootable disk.
2. Use the `mkfs`(1M) command to make a `bfs`file system in the appropriate partition of the disk.
3. Mount the new boot file system.
4. Populate the new file system; that is, copy into the new `bfs` file system all the required bootable programs and data files used during the boot procedure. See "The `stand` and `boot` Partitions" in the "Machine Management" chapter for information about these files.

Mounting and Unmounting File Systems

For a file system to be available to users, it must be mounted. The root and /usr file systems are always mounted as part of the boot procedure. The /usr file system may be on the same disk device as the root file system. The mount command that makes these two file systems available is contained in start-up shell procedures.

The mount command causes the mounted disk device and the mounted-on directory (the "mount point") to be associated with certain other information (such as the FSType, the mount options used during the mount, and the time the mount was performed) in the file /etc/mnttab (see mnttab(4)). For example, the command

```
mount -F s5 /dev/dsk/c1d0s2 /usr
```

tells the system to mount /dev/dsk/c1d0s2 as an s5 file system that begins at the directory /usr, while the command

```
mount -F ufs /dev/dsk/c1d0s2 /usr
```

tells the system to mount /dev/dsk/c1d0s2 as a ufs file system that begins at the directory /usr.

If you try to change directories (cd(1)) to a directory in the /usr file system before the mount command is issued, the cd command will fail. Until the mount command completes, the system does not know about any of the directories in the /usr file system. True, there is a directory /usr (it must exist at the time the mount command is issued), but the file system below that remains inaccessible until the mount command completes.

It is common for small file systems to be contained completely on a single floppy diskette. A diskette can hold as many as 1422 512-byte blocks. You can define file systems on diskette and use them either for storage or for live access. Using a diskette for live access rather than a hard disk has the following two disadvantages; it is slower, and it ties up the diskette device.

It is common for users to copy a file system on a diskette to a directory on a hard disk. To do that, the file system must first be mounted. A user who plans to establish a file system that can be brought in from diskettes needs first to create two directories on the hard disk; one to serve as a mount point and one to be the root directory of the file system being brought in. Assume that you have created the appropriate directories. The mount point is named /mnt, and the root is named /myfs. You can then copy an s5 file system from diskette to hard disk with the following command sequence:

```
mount -F s5 -o ro /dev/diskette /mnt

cd /mnt

find . -print | cpio -pdm /myfs
```

The -o option, followed by the argument ro, means read-only. See find(1) and cpio(1) for an explanation of the other options used.

The command for unmounting a file system requires only the name of the special device or the mount point. After you have copied in a file system from a diskette, for example, you can issue the command

```
umount /dev/diskette
```

to free the diskette drive.

Unmounting is frequently a first step before using other commands that operate on file systems. For example, fsck, which checks and repairs a file system, works on unmounted file systems. Unmounting is also an important part of the process of shutting the system down.

Maintaining a File System

Once a file system has been created and made available, it must be maintained regularly so that its performance remains satisfactory and so that it does not develop inconsistencies. The maintenance to ensure satisfactory performance will be dealt with in the rest of this section, that to ensure consistency in the next.

There are four tasks that should be part of routine maintenance if the administrator wants to be sure that the performance of the file system will be satisfactory. All of them are aimed at ensuring that disk space does not become so scarce that system performance is degraded. They are:

1. monitoring the percent of disk space used,
2. monitoring files and directories that grow,
3. identifying and removing inactive files, and
4. identifying large space users.

Monitoring the Percent of Disk Space Used

Monitoring disk space may be done at any time to see how close to capacity your system is running. Until a pattern has emerged, it is advisable to check every day. To do this, use the `df`(1M) command as follows:

```
df -t
```

The `-t` option causes the total allocated blocks and files to be displayed, as well as free blocks and files. When no file systems are named, information about all mounted file systems is displayed. If information on unmounted file systems is needed the file system name must be specified. For more information on the numerous options available to `df`, see the `df`(1M) manual page in the *System Administrator's Reference Manual.*

Monitoring Files and Directories that Grow

Almost any system that is used daily has several files and directories that grow through normal use. Some examples are:

File	Use
/var/adm/wtmp	history of system logins
/var/adm/sulog	history of su commands
/var/cron/log	history of actions of /usr/sbin/cron
/var/help/HELPLOG	actions of /usr/bin/help
/var/spell/spellhist	words that spell(1) fails to match

The frequency with which you should check growing files depends on how active your system is and how critical the disk space problem is. A good way to limit the size of such files is the following:

```
tail -50 /var/adm/sulog > /var/tmp/sulog

mv /var/tmp/sulog /var/adm/sulog
```

This sequence puts the last 50 lines of /var/adm/sulog into a temporary file, and then it moves the temporary file to /var/adm/sulog, thus truncating the file to the 50 most recent entries.

Identifying and Removing Inactive Files

Part of the job of cleaning up heavily loaded file systems involves locating and removing files that have not been used recently. The find(1) command locates files that have not been accessed recently. find searches a directory tree beginning at a point named on the command line. It looks for filenames that match a given set of expressions, and when a match is found, performs a specified action on the file.

```
find /home -type f -mtime +60 -print > \
   /var/tmp/deadfiles &
```

(The above command line is shown on two lines for readability.)

Here is what the example shows:

`/home`	specifies the pathname where `find` is to start. Presumably, your machine is organized in such a way that inactive user files will not often be found in the `root` file system.
`-type f`	tells `find` to look only for regular files, and to ignore special files, directories, and pipes.
`-mtime +60`	says you are interested only in files that have not been modified in 60 days.
`-print`	means that when a file is found that matches the `-type` and `-mtime` expressions, you want the pathname to be printed.
`> /var/tmp/deadfiles &`	directs the output to a temporary file and indicates that the process is to run in the background. This is a sensible precaution if your experience tells you to expect a substantial amount of output.

Identifying Large Space Users

Two commands provide useful information: du(1M) and `find(1)`.

du produces a summary of the block counts for files or directories named in the command line. For example:

```
du /home
```

displays the block count for all directories in the `/home` file system. Optional arguments allow you to refine the output somewhat. For example, `du -s` may be run against each user's login to monitor individual users.

The `find` command can be used to locate specific files that exceed a given size limit.

```
find /home -size +10 -print
```

This example produces a display of the pathnames of all files (and directories) in the `/home` file system that are larger than 10 (512-byte) blocks.

Quotas

The quota system is built around limits on the two principal resources of a file system: inodes and data blocks. For each of these resources, users may be assigned quotas. A quota in this case consists of two limits, known as the soft and hard limits. The hard limit represents an absolute limit on the resource, blocks or inodes, that the user may never exceed under any circumstances. Associated with the soft limit is a time limit set by the administrator. Users may exceed the soft limits assigned to them, but only for a limited amount of time- the time limit set by the administrator. This allows the user to temporarily exceed limits if needed, as long as they are back under those limits before the time limit expires. An example of such a situation might be the generation of a large file that is then printed and deleted.

In summary, for each user, you can assign quotas (soft and hard limits) for both blocks and inodes. You can also define a time limit that applies to all users on a file system indicating how long they can exceed the soft limits. There are actually two time limits: one for blocks, and one for inodes. You may define different time limits for different file systems. Also, users may have different quotas set on different file systems.

Using Quotas

Before turning quotas on for a file system for the first time:

- If the quotas are for a file system listed in `/etc/vfstab`, enter an "rq" in the `mntopts` field for that file system. If there is an "rw" in that field in the table, it should be replaced by "rq".
- Mount the file system and `cd` to the mount point. Create a file called `quotas`. This file should be owned by `root`, and not writable by others.
- Execute `edquota -t` to change the time limits for exceeding the soft limits for blocks owned, and/or inodes owned. These limits are initially set to the values defined for `DQ_FTIMELIMIT` and `DQ_BTIMELIMIT` in `/usr/include/sys/fs/ufs_quota.h` - normally 1 week. If you leave either time limit (the one for exceeding the block limit or the one for exceeding the inode limit) at 0, or if you set either limit to 0, the default values will apply. You can, of course, change them to something else.

- Execute edquota, with or without the -p option, to set user quotas. Once you have set the quotas for a user, you can use the -p option to set the same quotas for another user. Note that because you are not limited to UIDs that are already being used, you may set quotas for future users.

Before turning on quotas on a file system, always run quotacheck on the file system. This will sync up the quotas with the actual state of the file system, so that if the file system has been used since the last time the quotas were turned on, all of the quotas will be updated to reflect the current state. This also provides a sanity check on the quotas file.

Use quotaon to turn quotas on, and quotaoff to turn them off. If you use the -a option with either, the command will execute the desired action on each ufs file system with "rq" in the mntopts field of its vfstab entry. Otherwise, you must invoke the command on each individual file system.

To report on quotas an administrator can use repquota or quot to get information on all users on a file system, or use quota to get information on a single user. Normal users can use quota to get information on their own quotas; they cannot get information on anyone else's quotas.

The Effects of Quotas on the User

The following are the major effects of the use of quotas on users:

- If a user exceeds his/her soft limit for blocks or inodes, the timer is started. If the user then reduces usage to a level under the soft limit, the timer is turned off and all is well. But if the user still has not reduced usage to an appropriate level when the timer expires, any further attempts by the user to acquire more file system resources will fail and the user will receive error messages saying that the file system is full. These messages will persist until the user has reduced usage to a level below the soft limit.

- If a user tries to exceed the hard limit at any time, the attempt will fail and the utility will indicate that it has run out of space.

- Because no warning is given when the user has exceeded the soft limit, users should be advised to run quota frequently. Users should be encouraged to include quota in their .profile so that it runs when they log in.

Checking a File System for Consistency

When the UNIX operating system is brought up, a consistency check of the file systems should always be done. Often this check is done automatically as part of the power-up process. Included as part of that process is a sanity check of each file system on the hard disk using the `-m` option to `fsck`. The sanity check returns a code for each file system indicating whether the consistency checking and repair program, `fsck`(1M), should be run.

`fsck` should be used to check file systems not mounted routinely as part of the power-up process. If inconsistencies are discovered, corrective action must be taken before the file systems are mounted. The remainder of this section is designed to acquaint you with the command line options of the `fsck` utility, the type of checking it does in each of its phases, and the repairs it suggests.

It should be said at the outset that file system corruption, while serious, is not all that common. Most of the time a check of the file systems finds everything all right. The reason we put so much emphasis on file system checking is that if errors are allowed to go undetected, the loss can be substantial.

The `fsck` Utility

The file system check (`fsck`) utility is an interactive file system check and repair program. `fsck` uses the information contained in the file system to perform consistency checks. If an inconsistency is detected, a message describing the inconsistency is displayed. At that point you may decide whether to have `fsck` ignore the inconsistency or attempt to fix it. Reasons you might choose to have `fsck` ignore an inconsistency are that you think the problem is so severe that you want to fix it yourself, or that you plan to go back to an earlier version of the file system. Whatever your decision, you should not ignore the inconsistencies `fsck` reports. File system inconsistencies do not repair themselves. If they are ignored, they get worse.

The `fsck` administrative command is used to run the `fsck` utility to check and repair inconsistencies in a file system. With the exception of the `root` file system, a file system should be unmounted while it is being checked. If this is not possible, care should be taken that the system is quiescent and that it is rebooted immediately afterwards if the file system checked is a critical one.

The `root` file system should be checked only when the computer is in run level S and no other activity is taking place in the machine. The system should be rebooted immediately afterwards.

The generic format of the `fsck` command follows:

> `fsck` [`-F` *FSType*] [`-V`] [`-m`] [*special* ...]
>
> `fsck` [`-F` *FSType*] [`-V`] [*current_options*] [`-o` *specific_options*] [*special* ...]

The `-F` is used to specify the *FSType* on which the command must act. The *FSType* must be specified on the command line or must be determinable from `/etc/vfstab` by matching an entry in that file with the *special* specified.

The `-V` option causes the command to echo the completed command line. The echoed line will include additional information derived from `/etc/vfstab`. This option can be used to verify and validate the command line. It does not cause the command to execute.

The `-m` is used to perform a sanity check only. This option is usually used before mounting file systems because it lets the administrator know whether the file system needs to be checked.

current_options are options supported by the `s5`-specific module of the *command.*

The `-o` option is used to specify *FSType*-specific options if any. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

If the file system is inconsistent the user is prompted for concurrence before each correction is attempted. It should be noted that some corrective actions will result in some loss of data. The amount and severity of data loss may be determined from the diagnostic output. The default action for each correction is to wait for the user to respond `yes` or no. If the user does not have write permission `fsck` defaults to a no action.

In the rest of this chapter we will see how to use `fsck` to check `s5` and `ufs` and `bfs` file systems. The information is presented separately for each FSType. Although this results in a certain amount of repetition, it is hoped that it will avoid confusion.

Checking s5 File Systems

The following is the s5 specific format of the fsck command:

```
fsck [-F s5] [generic_options] [special...]

fsck [-F s5] [generic_options] [-y] [-n] [-p] [-sX] [-SX] [-tfile] \
[-l] [-q] [-D] [-f] [special...]
```

(The second command line is shown on two lines for readability.)

The options are as follows:

-y	Specifies a "yes" response for all questions. This is the normal choice when the command is being run as part of a shell procedure. It generally causes fsck to correct all errors.
-n	Specifies a "no" response for all questions. fsck will not write the file system.
-p	This option causes fsck to correct any innocuous inconsistencies. Unreferenced blocks, misaligned directories, incorrect link counts and bad free lists are some examples of inconsistencies which are automatically corrected.
-s*X*	Specifies an unconditional reconstruction of the free list. The *X* argument specifies the number of blocks-per-cylinder and the number of blocks to skip (rotational gap). The default values are those specified when the file system was created. The formats for some common disk drives are as follows for 1K file systems. See the "Using mkfs for s5" section of this chapter for more information.

Device	*-sblocks/cylinder:gap*
72M Hard Disk	`-s162:10`
30M Hard Disk	`-s90:10`
10M Hard Disk	`-s72:10`
Diskette	`-s18:1`

`-s`*X*	Specifies a conditional reconstruction of the free list, to be done only if corruption is detected. The format of the *X* argument is the same as described above for the `-s` option.
`-t`*file*	Specifies a scratch file for use in case the file system check requires additional memory. If this option is not specified, the process asks for a filename when more memory is needed.
`-l`	This option causes damaged files to be identified by their logical names in addition to the inode numbers.
`-q`	Specifies a "quiet" file system check. Output messages from the process are suppressed.
`-D`	Checks directories for bad blocks. This option is used to check file systems for damage after a system crash.
`-f`	Specifies that a fast file system check be done. Only Phase 1 (check blocks and sizes) and Phase 5 (check free list) are executed for a fast check. Phase 6 (reconstruct free list) is run only if necessary.
special	Names the special device file associated with a file system. If no device name is specified, `fsck` checks all file systems named in `/etc/vfstab` with a numeric `fsckpass` field.

Sample Command Use

The command line below shows `fsck` being entered to check the `usr` file system. No options are specified. The system response means that no inconsistencies were detected. The command operates in phases, some of which are run only if required or in response to a command line option. As each phase is completed, a message is displayed. At the end of the program a summary message is displayed showing the number of files (inodes) used, blocks used, and free blocks.

```
fsck -F s5 /dev/rdsk/c1d0s2

/dev/rdsk/c1d0s2
File System: usr Volume: usr

** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Free List
289 files 6522 blocks 3220 free
```

`s5` File System Components Checked by `fsck`

Before describing the phases of `fsck` and the messages that may appear in each, we will review the components of an `s5` file system and describe the kinds of consistency checks that are applied to each.

Super-Block

Every change to the file system blocks or inodes modifies the super-block. If the CPU is halted, and the last command involving output to the file system is not a `sync` command, the super-block will almost certainly be corrupted. The super-block can be checked for inconsistencies involving:

- file system size
- inode list size
- free-block list
- free-block count
- free inode count

File System Size and I-Node List Size

The number of blocks in a file system must be greater than the number of blocks used by the super-block plus the number of blocks used by the inode list. The number of inodes must be less than the maximum number allowed for the file system type. While there is no way to check these sizes precisely, `fsck` can check that they are within reasonable bounds. All other checks of the file system depend on the reasonableness of these values.

Free-Block List

The free-block list starts in the super-block and continues through the free-list blocks of the file system. Each free-list block can be checked for

- list count out of range
- block numbers out of range
- blocks already allocated within the file system

A check is made to see that all the blocks in the file system were found.

The first free-block list is in the super-block. The `fsck` program checks the list count for a value less than 0 or greater than 50. It also checks each block number to make sure it is within the range bounded by the first and last data block in the file system. Each block number is compared to a list of previously allocated blocks. If the free-list block pointer is not 0, the next free-list block is read and the process is repeated.

When all the blocks have been accounted for, a check is made to see if the number of blocks in the free-block list plus the number of blocks claimed by the inodes equals the total number of blocks in the file system. If anything is wrong with the free-block list, `fsck` can rebuild it leaving out blocks already allocated.

Free-Block Count

The super-block contains a count of the total number of free blocks within the file system. The `fsck` program compares this count to the number of blocks it finds free within the file system. If the counts do not agree, `fsck` can replace the count in the super-block by the actual free-block count.

Free I-Node Count

The super-block contains a count of the number of free inodes within the file system. The `fsck` program compares this count to the number of inodes it found free within the file system. If the counts do not agree, `fsck` can replace the count in the super-block by the actual free inode count.

I-Nodes

The list of inodes is checked sequentially starting with inode 1 (there is no inode 0). Each inode is checked for inconsistencies involving

- format and type
- link count
- duplicate blocks
- bad block numbers
- inode size

Format and Type

Each inode contains a mode word. This mode word describes the type and state of the inode. Inodes may be one of six types:

- regular
- directory
- block special
- character special
- FIFO (named-pipe)
- symbolic link

Inodes may be in one of three states: unallocated, allocated, and partially allocated. This last state means that the inode is incorrectly formatted. An inode can get into this state if, for example, bad data are written into the inode list because of a hardware failure. The only corrective action `fsck` can take is to clear the inode.

Link Count

Each inode contains a count of the number of directory entries linked to it. The `fsck` program verifies the link count of each inode by examining the entire directory structure, starting from the root directory, and calculating an actual link count for each inode.

Discrepancies between the link count stored in the inode and the actual link count as determined by `fsck` may be of three types:

- the stored count is not 0, the actual count is 0

  This can occur if no directory entry appears for the inode. In this case `fsck` can link the disconnected file to the `lost+found` directory.

- the stored count is not 0, the actual count is not 0, but the counts are unequal

  This can occur if a directory entry has been removed but the inode has not been updated. In this case `fsck` can replace the stored link count by the actual link count.

- the stored count is 0, the actual count is not 0

  In this case `fsck` can change the link count of inode to the actual count.

Duplicate Blocks

Each inode contains a list of all the blocks claimed by the inode. The `fsck` program compares each block number claimed by an inode to a list of allocated blocks. If a block number claimed by an inode is on the list of allocated blocks, it is put on a list of duplicate blocks. If it is not on the list of allocated blocks, it is put on it. If this process produces a list of duplicate blocks, `fsck` makes a second pass of the inode list to find the other inode that claims each duplicate block. (A large number of duplicate blocks in an inode may be caused by an indirect block not being written to the file system.) Although it is not possible to determine with certainty which inode is in error, in most cases the inode with the most recent modification time is correct. The `fsck` program prompts the user to clear both inodes.

Bad Block Numbers

The `fsck` program checks each block number claimed by an inode to see that its value is higher than that of the first data block and lower than that of the last block in the file system. If the block number is outside this range, it is considered a bad block number.

Bad block numbers in an inode may be caused by an indirect block not being written to the file system. The `fsck` program prompts the user to clear the inode.

NOTE A bad block number in a file system is not the same as a bad (that is, unreadable) block on a hard disk.

I-Node Size

Each inode contains a 32-bit (4-byte) size field. This field shows the number of characters in the file associated with the inode. A directory inode within the file system has the directory bit set in the inode mode word.

If the directory size is not a multiple of 16, `fsck` warns of directory misalignment and prompts for corrective action.

For a regular file, a rough check of the consistency of the size field of an inode can be performed by using the number of characters shown in the size field to calculate how many blocks should be associated with the inode, and comparing that to the actual number of blocks claimed by the inode.

Indirect Blocks

Indirect blocks are owned by an inode. Therefore, inconsistencies in an indirect block directly affect the inode that owns it. Inconsistencies that can be checked are:

- blocks already claimed by another inode
- block numbers outside the range of the file system

The consistency checks for direct blocks described in the sections "Duplicate Blocks" and "Bad Block Numbers" above are also performed for indirect blocks.

Directory Data Blocks

Directories are distinguished from regular files by an entry in the mode field of the inode. Data blocks associated with a directory contain the directory entries. Directory data blocks are checked for inconsistencies involving:

- directory inode numbers pointing to unallocated inodes
- directory inode numbers greater than the number of inodes in the file system
- incorrect directory inode numbers for "." and ". ." directories
- directories disconnected from the file system

Directory Unallocated

If a directory entry inode number points to an unallocated inode, `fsck` can remove that directory entry. This condition occurs if the data blocks containing the directory entries are modified and written out while the inode is not yet written out.

Bad I-Node Number

If a directory entry inode number is pointing beyond the end of the inode list, `fsck` can remove that directory entry. This condition occurs if bad data are written into a directory data block.

Incorrect "." and ".." Entries

The directory inode number entry for "." should be the first entry in the directory data block. Its value should be equal to the inode number for the directory data block.

The directory inode number entry for ". ." should be the second entry in the directory data block. Its value should be equal to the inode number for the parent of the directory entry (or the inode number of the directory data block if the directory is the root directory). If the directory inode numbers for "." and ". ." are incorrect, `fsck` can replace them with the correct values.

Disconnected Directories

The `fsck` program checks the general connectivity of the file system. If a directory is found that is not linked to the file system, `fsck` links the directory to the `lost+found` directory of the file system. (This condition can occur when inodes are written to the file system but the corresponding directory data blocks are not.) When a file is linked to the `lost+found` directory, the owner of the file must be notified.

Regular Data Blocks

Data blocks associated with a regular file hold the file's contents. `fsck` does not attempt to check the validity of the contents of a regular file's data blocks.

Running `fsck` on an `s5` File System

The `fsck` program runs in phases. Each phase reports any errors that it detects. Figure 5-12 lists the abbreviations that are used in the `fsck` error messages. If the error is one that `fsck` can correct, the user is asked if the correction should be made. This section describes the messages that are produced by each phase.

Figure 5-12: Error Message Abbreviations in `fsck` Output

The following abbreviations that appear in the messages have the meaning indicated by the text following them:

`BLK`	block number
`DUP`	duplicate block number
`DIR`	directory name
`MTIME`	time file was last modified
`UNREF`	unreferenced
`CG`	cylinder group

The following single-letter abbreviations are replaced by the text opposite them when the message appears on your screen:

`B`	block number
`F`	file (or directory) name
`I`	inode number
`M`	file mode
`O`	user-ID of a file's owner
`S`	file size
`T`	time file was last modified
`X`	link count,
or	number of BAD, DUP, or MISSING blocks
or	number of files (depending on context)
`Y`	corrected link count number
or	number of blocks in file system (depending on context)
`Z`	number of free blocks

Initialization Phase

Command line syntax is checked. Before the file system check can be performed, `fsck` sets up some tables and opens some files. The `fsck` program terminates when it encounters errors during the initialization phase.

General Errors

The following three error messages may appear in any phase after initialization. While they offer the option to continue, it is generally best to regard them as fatal, end the run, and try to determine what caused the problem.

Message

```
CAN NOT SEEK: BLK B (CONTINUE?)
```

A request to move to a specified block number *B* in the file system failed. This message indicates a serious problem, probably a hardware failure.

Message

```
CAN NOT READ: BLK B (CONTINUE?)
```

A request to read a specified block number *B* in the file system failed. The message indicates a serious problem, probably a hardware failure.

Message

```
CAN NOT WRITE: BLK B (CONTINUE?)
```

A request to write a specified block number *B* in the file system failed. The disk may be write-protected.

Meaning of Yes/No Responses

An n(no) response to the `CONTINUE?` prompt means:

Terminate the program.
(This is the recommended response.)

A y(yes) response to the `CONTINUE?` prompt means:

Attempt to continue to run the file system check.

Note that the problem will ofter recur. This error condition prevents a complete check of the file system. A second run of `fsck` should be made to recheck the file system.

Phase 1: Check Blocks and Sizes

This phase checks the inode list. It reports error conditions encountered while:

- checking inode types
- setting up the zero-link-count table
- examining inode block numbers for bad or duplicate blocks
- checking inode size
- checking inode format

Types of Error Messages - Phase 1

Phase 1 produces four types of error messages:

1. informational messages
2. messages with a `CONTINUE?` prompt
3. messages with a `CLEAR?` prompt
4. messages with a `RECOVER?` prompt

There is a connection between some informational messages and messages with a `CONTINUE?` prompt. The `CONTINUE?` prompt generally indicates that some limit has been reached.

Meaning of Yes/No Responses - Phase 1

An n(no) response to the `CONTINUE?` prompt means:

Terminate the program.

In Phase 1, a y(yes) response to the `CONTINUE?` prompt means:

Continue with the program.
When this error occurs a complete check of the file system is not possible. A second run of `fsck` should be made to recheck the file system.

An n(no) response to the `RECOVER?` prompt means:

Recover all the blocks to which the inode points.
A no response is only appropriate if the user intends to delete the excess blocks.

An n(no) response to the CLEAR? prompt means:

Ignore the error condition.
A no response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the CLEAR? prompt means:

Deallocate the inode *I* by zeroing out its contents.
This may generate the UNALLOCATED error condition in Phase 2 for each directory entry pointing to this inode.

Phase 1 Error Messages

Message

```
UNKNOWN FILE TYPE I= I (CLEAR?)
```

The mode word of the inode *I* indicates that the inode is not a pipe, special character inode, regular inode, or directory inode. If the –p option is specified the inode will be cleared.

Message

```
LINK COUNT TABLE OVERFLOW (CONTINUE?)
```

An internal table for fsck containing allocated inodes with a link count of zero has no more room. If the –p option is specified the program will exit and fsck will have to be completed manually.

Message

```
B BAD I= I
```

I-node *I* contains block number *B* with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system. This error condition may generate the `EXCESSIVE BAD BLKS` error message in Phase 1 if inode *I* has too many block numbers outside the file system range. This error condition generates the `BAD/DUP` error message in Phases 2 and 4.

Message

```
EXCESSIVE BAD BLOCKS I= I (CONTINUE?)
```

There are too many (usually more than 10) blocks with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system associated with inode *I*. If the -p option is specified the program will terminate.

Message

```
B DUP I= I
```

Inode *I* contains block number *B*, which is already claimed by the same or another inode or by a free-list. This error condition may generate the `EXCESSIVE DUP BLKS` error message in Phase 1 if inode *I* has too many block numbers claimed by the same or another inode or by a free-list. This error condition invokes Phase 1B and generates the `BAD/DUP` error messages in Phases 2 and 4.

Message

```
EXCESSIVE DUP BLKS I= I (CONTINUE?)
```

There are too many (usually more than 10) blocks claimed by the same or another inode or by a free-list. If the -p option is specified the program will terminate.

Message

```
DUP TABLE OVERFLOW (CONTINUE?)
```

An internal table in `fsck` containing duplicate block numbers has no more room. If the -p option is specified the program will terminate.

Message

```
DIRECTORY MISALIGNED I= I
```

The size of a directory inode is not a multiple of 16. If the -p option is used, the directory will be recovered automatically.

Message

```
PARTIALLY ALLOCATED INODE I= I (CLEAR?)
```

I-node *I* is neither allocated nor unallocated. If the -p option is specified the inode will be cleared.

Message

```
DIR/FILE SIZE ERROR
```

The file references more or less data than is indicated by the inode.

Message

```
DELETE OR RECOVER EXCESS DATA
```

The user has the choice of deleting or recovering the excess blocks pointed to by the inode.

Message

```
RECOVER?
```

The file references more data than is indicated by the inode. The user is given the choice of correcting the inode information. If the -p option is specified the data will be recovered.

Message

```
DELETE?
```

The file references more data than is indicated by the inode. The user is given the choice of deleting the referenced blocks and leaving the inode data intact.

Phase 1B: Rescan for More DUPS

When a duplicate block is found in the file system, the file system is rescanned to find the inode that previously claimed that block. When the duplicate block is found, the following informational message is printed:

Message

```
DUP I= I
```

Inode *I* contains block number *B* that is already claimed by the same or another inode or by a free-list. This error condition generates the BAD/DUP error message in Phase 2. Inodes that have overlapping blocks may be determined by examining this error condition and the DUP error condition in Phase 1.

Phase 2: Check Pathnames

This phase removes directory entries pointing to bad inodes found in Phases 1 and 1B. It reports error conditions resulting from

- incorrect root inode mode and status
- directory inode pointers out of range
- directory entries pointing to bad inodes

Types of Error Messages—Phase 2

Phase 2 has four types of error messages:

1. informational messages
2. messages with a FIX? prompt
3. messages with a CONTINUE? prompt
4. messages with a REMOVE? prompt

Meaning of Yes/No Responses—Phase 2

An n(no) response to the `FIX?` prompt means:

Terminate the program because `fsck` will be unable to continue.

A y(yes) response to the `FIX?` prompt means:

Change the root inode type to "directory."
If the root inode data blocks are not directory blocks, a very large number of error messages are generated.

An n(no) response to the `CONTINUE?` prompt means:

Terminate the program.

A y(yes) response to the `CONTINUE?` prompt means:

Ignore the DUPS/BAD IN ROOT INODE error message and continue to run the file system check.
If the root inode is not correct, a large number of other error messages may be generated.

An n(no) response to the `REMOVE?` prompt means:

Ignore the error condition.
A no response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `REMOVE?` prompt means:

Remove duplicate or unallocated blocks.

Phase 2 Error Messages

Message

```
ROOT INODE UNALLOCATED. TERMINATING
```

The root inode (usually inode number 2) of the file system has no allocate mode bits. This error message indicates a serious problem that causes the program to stop. Call your service representative.

Message

```
ROOT INODE NOT DIRECTORY (FIX?)
```

The root inode (usually inode number 2) of the file system is not directory inode type. If the -p option is specified the program will terminate.

Message

```
DUPS/BAD IN ROOT INODE (CONTINUE?)
```

Phase 1 or 1B found duplicate blocks or bad blocks in the root inode (usually inode number 2) of the file system. If the -p option is specified the program will terminate.

Message

```
I OUT OF RANGE I= I NAME= F (REMOVE?)
```

A directory entry *F* has an inode number *I* that is greater than the end of the inode list. If the -p option is specified the inode will be removed automatically.

Message

```
UNALLOCATED I= I OWNER= O MODE= M SIZE= S MTIME= T NAME= F (REMOVE?)
```

A directory entry *F* has an inode *I* without allocate mode bits. The owner *O*, mode *M*, size *S*, modify time *T*, and filename *F* are printed. If the file system is not mounted and the -n option was not specified, the entry is removed automatically if the inode it points to is character size 0. The entry is removed if the -p option is specified.

Message

```
DUP/BAD I= I OWNER= O MODE= M SIZE= S MTIME= T DIR= F (REMOVE?)
```

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with directory entry *F*, directory inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed. If the -p option is specified the duplicate/bad blocks are removed.

Message

```
DUP/BAD I= I OWNER= O MODE= M SIZE= S MTIME= T FILE= F (REMOVE?)
```

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with file entry *F*, inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and filename *F* are printed. If the -p option is specified the duplicate/bad blocks are removed.

Message

```
BAD BLK B IN DIR I= I OWNER= O MODE= M SIZE= S MTIME= T
```

This message only occurs when the -D option is used. A physically damaged block was found in directory inode *I*. Error conditions looked for in directory blocks are nonzero padded entries, inconsistent "." and ". ." entries, and embedded slashes in the name field. This error message means that the user should at a later time either remove the directory inode if the entire block looks bad or change (or remove) those directory entries that look bad.

Phase 3: Check Connectivity

This phase checks the directories examined in Phase 2. It reports error conditions resulting from

- unreferenced directories
- missing or full `lost+found` directories

Types of Error Messages—Phase 3

Phase 3 has two types of error messages:

1. informational messages
2. messages with a `RECONNECT?` prompt

Meaning of Yes/No Responses—Phase 3

An n(no) response to the `RECONNECT?` prompt means:

> Ignore the error condition.
> This response generates `UNREF` error messages in Phase 4.
> A no response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `RECONNECT?` prompt means:

> Reconnect directory inode *I* to the file system in the directory for lost files (usually the `lost+found` directory).
> This may generate `lost+found` error messages if there are problems connecting directory inode *I* to the `lost+found` directory. If the link is successful, a `CONNECTED` informational message appears.

Phase 3 Error Messages

Message

```
UNREF DIR I= I OWNER= O MODE= M SIZE= S MTIME= T (RECONNECT?)
```

The directory inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of directory inode *I* are printed. The `fsck` program forces the reconnection of a nonempty directory. If the `-p` option is specified the nonempty directory is reconnected.

Message

```
SORRY. NO lost+found DIRECTORY
```

There is no `lost+found` directory in the root directory of the file system; `fsck` ignores the request to link a directory to the `lost+found` directory. This generates the `UNREF` error message in Phase 4. The access modes of the `lost+found` directory may be incorrect.

Message

```
SORRY. NO SPACE IN lost+found DIRECTORY
```

There is no space to add another entry to the `lost+found` directory in the root directory of the file system; `fsck` ignores the request to link a directory to the `lost+found` directory. This generates the `UNREF` error message in Phase 4. Clear out unnecessary entries in the `lost+found` directory or make it larger.

Message

```
DIR I= I1 CONNECTED. PARENT WAS I= I2
```

This is an advisory message indicating a directory inode *I1* was successfully connected to the `lost+found` directory. The parent inode *I2* of the directory inode *I1* is replaced by the inode number of the `lost+found` directory.

Phase 4: Check Reference Counts

This phase checks the link count information obtained in Phases 2 and 3. It reports error conditions resulting from:

- unreferenced files
- a missing or full `lost+found` directory
- incorrect link counts for files, directories, or special files
- unreferenced files and directories
- bad or duplicate blocks in files and directories
- incorrect total free-inode counts

Types of Error Messages—Phase 4

Phase 4 has five types of error messages:

1. informational messages
2. messages with a `RECONNECT?` prompt
3. messages with a `CLEAR?` prompt
4. messages with an `ADJUST?` prompt
5. messages with a `FIX?` prompt

Meaning of Yes/No Responses—Phase 4

An n(no) response to the `RECONNECT?` prompt means:

Ignore this error condition.
This response generates a `CLEAR` error message later in Phase 4.

A y(yes) response to the `RECONNECT?` prompt means:

Reconnect inode *I* to the file system in the directory for lost files (usually the `lost+found` directory).
This can generate a `lost+found` error message in this phase if there are problems connecting inode *I* to the `lost+found` directory.

An n(no) response to the `CLEAR?` prompt means:

Ignore the error condition.
This response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `CLEAR?` prompt means:

Deallocate the inode by zeroing out its contents.

An n(no) response to the `ADJUST?` prompt means:
Ignore the error condition.
This response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `ADJUST?` prompt means:
Replace the link count of file inode *I* by *Y*.

An n(no) response to the `FIX?` prompt means:

Ignore the error condition.
This response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `FIX?` prompt means:

Replace the count in super-block by the actual count.

Phase 4 Error Messages

Message

```
UNREF FILE I= I OWNER= O MODE= M SIZE= S MTIME= T (RECONNECT?)
```

I-node *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the –n option is omitted and the file system is not mounted, empty files are cleared automatically. Nonempty files are not cleared. If the –p option is specified the inode is reconnected.

Message

```
SORRY, NO lost+found DIRECTORY
```

There is no lost+found directory in the root directory of the file system; fsck ignores the request to link a file to the lost+found directory. This generates the CLEAR error message later in Phase 4. The access modes of the lost+found directory may be incorrect.

Message

```
SORRY, NO SPACE IN lost+found DIRECTORY
```

There is no space to add another entry to the lost+found directory in the root directory of the file system; fsck ignores the request to link a file to the lost+found directory. This generates the CLEAR error message later in Phase 4. Check the size and contents of the lost+found directory.

Message

```
(CLEAR)
```

The inode mentioned in the UNREF error message immediately preceding cannot be reconnected.

Message

```
LINK COUNT FILE I= I OWNER= O MODE= M SIZE= S MTIME= T COUNT= X SHOULD BE Y (ADJUST?)
```

The link count for file inode *I*, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* are printed. If the -p option is specified the link count is adjusted.

Message

```
LINK COUNT DIR I= I OWNER= O MODE= M SIZE= S MTIME= T COUNT= X SHOULD BE Y (ADJUST?)
```

The link count for directory inode *I*, is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the -p option is specified the link count is adjusted.

Message

```
UNREF FILE I= I OWNER= O MODE= M SIZE= S MTIME= T (CLEAR?)
```

File inode *I*, was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the -n option is omitted and the file system is not mounted, empty files are cleared automatically. Nonempty directories are not cleared. If the -p option is specified the file is cleared if it can not be reconnected.

Message

```
UNREF DIR I= I OWNER= O MODE= M SIZE= S MTIME= T (CLEAR?)
```

Directory inode *I*, was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the -n option is omitted and the file system is not mounted, empty directories are cleared automatically. Nonempty directories are not cleared. If the -p option is specified the directory is cleared if it can not be reconnected.

Message

```
BAD/DUP FILE I= I OWNER= O MODE= M SIZE= S MTIME= T (CLEAR?)
```

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with file inode *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the –p option is specified the file is cleared.

Message

```
BAD/DUP DIR I= I OWNER= O MODE= M SIZE= S MTIME= T (CLEAR?)
```

Phase 1 or Phase 1B found duplicate blocks or bad blocks associated with directory inode *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. If the –p option is specified the directory is cleared.

Message

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

The actual count of the free inodes does not match the count in the super-block of the file system. If the –q or –p option is specified, the count in the super-block will be fixed automatically.

Phase 5: Check Free List

This phase checks the free-block list. It reports error conditions resulting from:

- bad blocks in the free-block list
- a bad free-block count
- duplicate blocks in the free-block list

- unused blocks from the file system that are not in the free-block list
- an incorrect total free-block count

Types of Error Messages - Phase 5

Phase 5 has four types of error messages:

1. informational messages
2. messages that have a `CONTINUE?` prompt
3. messages that have a `FIX?` prompt
4. messages that have a `SALVAGE?` prompt

Meaning of Yes/No Responses - Phase 5

An n(no) response to the `CONTINUE?` prompt means:

Terminate the program.

A y(yes) response to the `CONTINUE?` prompt means:

Ignore the rest of the free-block list and continue the execution of `fsck`.
This generates the `BAD BLKS IN FREE LIST` error message later in Phase 5.

An n(no) response to the `FIX?` prompt means:

Ignore the error condition.
This response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `FIX?` prompt means:

Replace the count in the super-block by the actual count.

An n(no) response to the `SALVAGE?` prompt means:

Ignore the error condition.
This response is only appropriate if the user intends to take other measures to fix the problem.

A y(yes) response to the `SALVAGE?` prompt means:

Replace the actual free-block list by a new free-block list.
The new free-block list will be ordered according to the gap and

cylinder specifications of the -s or -S option to reduce the time spent waiting for the disk to rotate into position.

Phase 5 Error Messages

Message

```
EXCESSIVE BAD BLKS IN FREE LIST (CONTINUE?)
```

The free-block list contains too many blocks with a value less than the first data block in the file system or greater than the last block in the file system. If the -p option is specified the program terminates.

Message

```
EXCESSIVE DUP BLKS IN FREE LIST (CONTINUE?)
```

The free-block list contains too many blocks claimed by inodes or earlier parts of the free block list. If the -p option is specified the program terminates.

Message

```
BAD FREEBLK COUNT
```

The count of free blocks in a free-list block is greater than 50 or less than 0. This condition generates the BAD FREE LIST message later in Phase 5.

Message

```
X BAD BLKS IN FREE LIST
```

X blocks in the free-block list have a block number lower than the first data block in the file system or greater than the last block in the file system. This condition generates the `BAD FREE LIST` message later in Phase 5.

Message

```
X DUP BLKS IN FREE LIST
```

X blocks claimed by inodes or earlier parts of the free-list block were found in the free-block list. This condition generates the `BAD FREE LIST` message later in Phase 5.

Message

```
X BLK(S) MISSING
```

X blocks unused by the file system were not found in the free-block list. This condition generates the `BAD FREE LIST` message later in Phase 5.

Message

```
FREE BLK COUNT WRONG IN SUPERBLOCK (FIX?)
```

The actual count of free blocks does not match the count of free blocks in the super-block of the file system. If the `-p` option was specified, the free block count in the super-block is fixed automatically.

Message

```
BAD FREE LIST (SALVAGE?)
```

This message is always preceded by one or more of the Phase 5 informational messages. If the -q or -p option was specified, the free-block list will be salvaged automatically.

Phase 6: Salvage Free List

This phase reconstructs the free-block list. It may display an advisory message about the blocks-to-skip or blocks-per-cylinder values.

Phase 6 Error Messages

Message

```
DEFAULT FREE-BLOCK LIST SPACING ASSUMED
```

This is an advisory message indicating that the blocks-to-skip (gap) is greater than the blocks-per-cylinder, the blocks-to-skip is less than 1, the blocks-per-cylinder is less than 1, or the blocks-per-cylinder is greater than 500. The default values of 10 blocks-to-skip and 162 blocks-per-cylinder are used.

NOTE Because the default values used may not be accurate for your system, care must be taken to specify correct values with the -s option on the command line. See the `fsck`(1M) and `mkfs`(1M) manual pages for further details.

Cleanup Phase

Once a file system has been checked, a few cleanup functions are performed. The cleanup phase displays advisory messages about the file system and the status of the file system.

Cleanup Phase Messages

Message

```
X files Y blocks Z free
```

This is an advisory message indicating that the file system checked contained *X* files using *Y* blocks, and that there are *Z* blocks free in the file system.

Message

```
***** FILE SYSTEM WAS MODIFIED *****
```

This is an advisory message indicating that the file system was modified by `fsck`.

Checking `ufs` File Systems

This section describes the use of `fsck` with `ufs` file systems. It assumes that you are running `fsck` interactively, and that all possible errors can be encountered. When an inconsistency is discovered in this mode, `fsck` reports the inconsistency for you to choose a corrective action. (You can also run `fsck` automatically using the `-y`, `-n`, or `-o p` options.)

`ufs` File System Components Checked by `fsck`

Before describing the phases of `fsck` and the messages that may appear in each, we will discuss the kinds of consistency checks applied to each component of a `ufs` file system.

Super-Block

The most commonly corrupted item in a file system is the summary information associated with the super-block, because it is modified with every change to the blocks or inodes of the file system, and is usually corrupted after an unclean halt. The super-block is checked for inconsistencies involving:

- file system size
- number of inodes
- free block count
- free inode count

File System Size

The file system size must be larger than the number of blocks used by the super-block and the number of blocks used by the list of inodes. While there is no way to check these sizes precisely, `fsck` can check that they are within reasonable bounds. All other file system checks require that these sizes be correct. If `fsck` detects corruption in the static parameters of the default super-block, `fsck` requests the operator to specify the location of an alternate super-block.

Free Block List

`fsck` checks that all the blocks marked as free in the cylinder group block maps are not claimed by any files. When all the blocks have been initially accounted for, `fsck` checks that the number of free blocks plus the number of blocks claimed by the inodes equals the total number of blocks in the file system. If anything is wrong with the block allocation maps, `fsck` will rebuild them, based on the list it has computed of allocated blocks.

Free Block Count

The summary information associated with the super-block contains a count of the total number of free blocks within the file system. `fsck` compares this count to the number of free blocks it finds within the file system. If the two counts do not agree, then `fsck` replaces the incorrect count in the summary information by the actual free block count.

Free I-Node Count

The summary information contains a count of the total number of free inodes within the file system. `fsck` compares this count to the number of free inodes it finds within the file system. If the two counts do not agree, then `fsck` replaces the incorrect count in the summary information by the actual free inode count.

I-Nodes

The list of inodes in the file system is checked sequentially starting with inode 2 (inode 0 marks unused inodes; inode 1 is saved for future generations) and progressing through the last inode in the file system. Each inode is checked for inconsistencies involving:

- format and type
- link count
- duplicate blocks
- bad block numbers
- inode size

Format and Type

Each inode contains a mode word. This mode word describes the type and state of the inode. Inodes may be one of six types: regular, directory, symbolic link, special block, special character, or named-pipe. Inodes may be in one of three allocation states: unallocated, allocated, and neither unallocated nor allocated. This last state means that the inode is incorrectly formatted. An inode can get into this state if bad data are written into the inode list. The only possible corrective action `fsck` can take is to clear the inode.

Link Count

Each inode counts the total number of directory entries linked to the inode. `fsck` verifies the link count of each inode by starting at the root of the file system, and descending through the directory structure. The actual link count for each inode is calculated during the descent.

If the stored link count is non-zero and the actual link count is zero, then no directory entry appears for the inode. If this happens, `fsck` will place the disconnected file in the `lost+found` directory. If the stored and actual link counts are non-zero and unequal, a directory entry may have been added or removed without the inode being updated. If this happens, `fsck` replaces the incorrect stored link count by the actual link count.

Each inode contains a list, or pointers to lists (indirect blocks), of all the blocks claimed by the inode. Since indirect blocks are owned by an inode, inconsistencies in an indirect block directly affect the inode that owns it.

Duplicate Blocks

`fsck` compares each block number claimed by an inode against a list of already allocated blocks. If another inode already claims a block number, then the block number is added to a list of duplicate blocks. Otherwise, the list of allocated blocks is updated to include the block number.

If there are any duplicate blocks, `fsck` performs a partial second pass over the inode list to find the inode of the duplicated block. The second pass is needed, since without examining the files associated with these inodes for correct content, not enough information is available to determine which inode is corrupted and should be cleared. If this condition does arise, then the inode with the earliest modify time is usually incorrect, and should be cleared. If this happens, `fsck` prompts the operator to clear both inodes. The operator must decide which one should be kept and which one should be cleared.

Bad Block Numbers

`fsck` checks the range of each block number claimed by an inode. If the block number is lower than the first data block in the file system, or greater than the last data block, then the block number is a bad block number. Many bad blocks in an inode are usually caused by an indirect block that was not written to the file system, a condition which can only occur if there has been a hardware failure. If an inode contains bad block numbers, `fsck` prompts the operator to clear it.

Inode Size

Each inode contains a count of the number of data blocks that it contains. The number of actual data blocks is the sum of the allocated data blocks and the indirect blocks. `fsck` computes the actual number of data blocks and compares that block count against the actual number of blocks the inode claims. If an inode contains an incorrect count `fsck` prompts the operator to fix it.

Each inode contains a thirty-two bit size field. The size is the number of data bytes in the file associated with the inode. The consistency of the byte size field is roughly checked by computing from the size field the maximum number of blocks that should be associated with the inode, and comparing that expected block count against the actual number of blocks the inode claims.

Data Associated with an I-Node

An inode can directly or indirectly reference three kinds of data blocks. All referenced blocks must be the same kind. The three types of data blocks are: plain data blocks, symbolic link data blocks, and directory data blocks. Plain data blocks contain the information stored in a file; symbolic link data blocks contain the path name stored in a link. Directory data blocks contain directory entries. `fsck` can only check the validity of directory data blocks.

Directory Data Blocks

Directory data blocks are checked for inconsistencies involving:

- directory inode numbers pointing to unallocated inodes
- directory inode numbers that are greater than the number of inodes in the file system
- incorrect directory inode numbers for "." and ".."
- directories that are not attached to the file system

Directory Unallocated

If the inode number in a directory data block references an unallocated inode, then `fsck` will remove that directory entry.

Bad I-Node Number

If a directory entry inode number references outside the inode list, then `fsck` will remove that directory entry. This condition occurs if bad data are written into a directory data block.

Incorrect "." and ".." Entries

The directory inode number entry for "`.`" must be the first entry in the directory data block. The inode number for "`.`" must reference itself; e.g., it must equal the inode number for the directory data block. The directory inode number entry for "`..`" must be the second entry in the directory data block. Its value must equal the inode number for the parent of the directory entry (or the inode number of the directory data block if the directory is the root directory). If the directory inode numbers are incorrect, `fsck` will replace them with the correct values. If there are multiple hard links to a directory, the first one encountered is considered the real parent to which "`..`" should point; `fsck` recommends deletion for the subsequently discovered names.

Disconnected Directories

`fsck` checks the general connectivity of the file system. If directories are not linked into the file system, then `fsck` links the directory back into the file system in the `lost+found` directory.

Running `fsck` on a `ufs` File System

`fsck` is a multi-pass file system check program. Each file system pass invokes a different phase of the `fsck` program. After initialization, `fsck` performs successive passes over each file system, checking blocks and sizes, path names, connectivity, reference counts, and the map of free blocks (possibly rebuilding it), and performs some cleanup.

At boot time `fsck` is normally run with the `-y` option, non-interactively. (`fsck` can also be run interactively by the administrator at any time.) `fsck` can also be run non-interactively to "preen" the file systems after an unclean halt. While preening a file system, it will only fix corruptions that are expected to result from an unclean halt. These actions are a subset of the actions that `fsck` takes when it is running interactively. When an inconsistency is detected, `fsck` generates an error message. If a response is required, `fsck` prints a prompt and waits for a response. When preening most errors are fatal. For those that are expected, the response taken is noted. This section explains the meaning of each error message, the possible responses, and the related error conditions.

The error conditions are organized by the phase of the `fsck` program in which they can occur. The error conditions that may occur in more than one phase are discussed under initialization.

Initialization Phase

Before a file system check can be performed, certain tables have to be set up and certain files opened. The messages in this section relate to error conditions resulting from command line options, memory requests, the opening of files, the status of files, file system size checks, and the creation of the scratch file.

Message

```
cannot alloc NNN bytes for blockmap
cannot alloc NNN bytes for freemap
cannot alloc NNN bytes for statemap
cannot alloc NNN bytes for lncntp
```

`fsck`'s request for memory for its virtual memory tables failed. This should never happen. When it does, `fsck` terminates. This is a serious system failure and should be handled immediately. Contact your service representative or another qualified person.

Message

```
Can't open checklist file: F
```

The file system checklist or default file *F* (usually `/etc/vfstab`) cannot be opened for reading. When this occurs, `fsck` terminates. Check the access modes of *F*.

Message

```
Can't stat root
```

fsck's request for statistics about the root directory failed. This should never happen. When it does, fsck terminates. Contact your service representative or another qualified person.

Message

```
Can't stat F
Can't make sense out of name F
```

fsck's request for statistics about the file system *F* failed. When running interactively, it ignores this file system and continues checking the next file system given. Check the access modes of *F*.

Message

```
Can't open F
```

fsck's attempt to open the file system *F* failed. When running interactively, it ignores this file system and continues checking the next file system given. Check the access modes of *F*.

Message

```
F: (NO WRITE)
```

Either the -n flag was specified or fsck's attempt to open the file system *F* for writing failed. When fsck is running interactively, all the diagnostics are printed out, but fsck does not attempt to fix anything.

Message

```
file is not a block or character device; OK
```

The user has given `fsck` the name of a regular file by mistake. Check the type of the file specified.

Possible responses to the `OK` prompt are:

YES	Ignore this error condition.
NO	Ignore this file system and continue checking the next file system given.

Message

```
UNDEFINED OPTIMIZATION IN SUPERBLOCK (SET TO DEFAULT)
```

The super-block optimization parameter is neither `OPT_TIME` nor `OPT_SPACE`.

Possible responses to the `SET TO DEFAULT` prompt are:

YES	Set the super-block to request optimization to minimize running time of the system. (If optimization to minimize disk space use is desired, it can be set using `tunefs`(1M).)
NO	Ignore this error condition.

Message

```
IMPOSSIBLE MINFREE=D IN SUPERBLOCK (SET TO DEFAULT)
```

The super-block minimum space percentage is greater than 99 percent or less then 0 percent.

Possible responses to the SET TO DEFAULT prompt are:

YES	Set the `minfree` parameter to 10 percent. (If some other percentage is desired, it can be set using `tunefs` (1M).)
NO	Ignore this error condition.

Message

```
MAGIC NUMBER WRONG
NCG OUT OF RANGE
CPG OUT OF RANGE
NCYL DOES NOT JIVE WITH NCG*CPG
SIZE PREPOSTEROUSLY LARGE
TRASHED VALUES IN SUPER BLOCK
```

followed by the message:

```
F: BAD SUPER BLOCK: B
USE -b OPTION TO FSCK TO SPECIFY LOCATION OF AN ALTERNATE
SUPER-BLOCK TO SUPPLY NEEDED INFORMATION; SEE fsck(1M).
```

The super-block has been corrupted. An alternative super-block must be selected from among the available copies. Choose an alternative super-block by calculating its offset or call your service representative or another qualified person. Specifying block 32 is a good first choice.

Message

```
INTERNAL INCONSISTENCY: M
```

`fsck` has had an internal panic, whose message is *M*. This should never happen. If it does, contact your service representative or another qualified person.

Message

```
CAN NOT SEEK: BLK B (CONTINUE)
```

`fsck`'s request to move to a specified block number *B* in the file system failed. This should never happen. If it does, contact your service representative or another qualified person.

Possible responses to the `CONTINUE` prompt are:

YES	Attempt to continue to run the file system check. (Note that the problem will often persist.) This error condition prevents a complete check of the file system. A second run of `fsck` should be made to recheck the file system. If the block was part of the virtual memory buffer cache, `fsck` will terminate with the message

```
Fatal I/O error
```

NO	Terminate the program.

Message

```
CAN NOT READ: BLK B (CONTINUE)
```

`fsck`'s request to read a specified block number *B* in the file system failed. This should never happen. If it does, contact your service representative or another qualified person.

Possible responses to the CONTINUE prompt are:

YES Attempt to continue to run the file system check. fsck will retry the read and print out the message:

```
THE FOLLOWING SECTORS COULD NOT BE READ: N
```

where *N* indicates the sectors that could not be read. If fsck ever tries to write back one of the blocks on which the read failed it will print the message:

```
WRITING ZERO'ED BLOCK N TO DISK
```

where *N* indicates the sector that was written with zero's. If the disk is experiencing hardware problems, the problem will persist. This error condition prevents a complete check of the file system. A second run of fsck should be made to recheck the file system. If the block was part of the virtual memory buffer cache, fsck will terminate with the message

```
Fatal I/O error
```

NO Terminate the program.

Message

```
CAN NOT WRITE: BLK B (CONTINUE)
```

fsck's request to write a specified block number *B* in the file system failed. The disk is write-protected; check the write-protect lock on the drive. If that is not the problem, contact your service representative or another qualified person.

Possible responses to the CONTINUE prompt are:

YES	Attempt to continue to run the file system check. The write operation will be retried. Sectors that could not be written will be indicated by the message:

```
THE FOLLOWING SECTORS COULD NOT BE WRITTEN: N
```

where *N* indicates the sectors that could not be written. If the disk is experiencing hardware problems, the problem will persist. This error condition prevents a complete check of the file system. A second run of fsck should be made to recheck this file system. If the block was part of the virtual memory buffer cache, fsck will terminate with the message

```
Fatal I/O error
```

NO	Terminate the program.

Message

```
bad inode number DDD to ginode
```

An internal error was caused by an attempt to read non-existent inode *DDD*. This error causes fsck to exit. If this occurs, contact your service representative or another qualified person.

Phase 1: Check Blocks and Sizes

This phase checks the inode list. It reports error conditions encountered while:

- checking inode types
- setting up the zero-link-count table
- examining inode block numbers for bad or duplicate blocks
- checking inode size
- checking inode format

All the errors in this phase except `INCORRECT BLOCK COUNT` and `PARTIALLY TRUNCATED INODE` are fatal if the file system is being preened.

Phase 1 Error Messages

Message

```
UNKNOWN FILE TYPE I=I (CLEAR)
```

The mode word of the inode *I* indicates that the inode is not a special block inode, special character inode, socket inode, regular inode, symbolic link, FIFO file, or directory inode.

Possible responses to the `CLEAR` prompt are:

YES	De-allocate inode *I* by zeroing out its contents. This will always generate the `UNALLOCATED` error message in Phase 2 for each directory entry pointing to this inode.
NO	Ignore this error condition.

Message

```
PARTIALLY TRUNCATED INODE I=I (SALVAGE)
```

`fsck` has found inode *I* whose size is shorter than the number of blocks allocated to it. This condition should only occur if the system crashes while truncating a file. When preening the file system, `fsck` completes the truncation to the specified size.

Possible responses to the `SALVAGE` prompt are:

YES	Complete the truncation to the size specified in the inode.
NO	Ignore this error condition.

Message

```
LINK COUNT TABLE OVERFLOW (CONTINUE)
```

An internal table for `fsck` containing allocated inodes with a link count of zero has no more room.

Possible responses to the `CONTINUE` prompt are:

YES	Continue with the program. This error condition prevents a complete check of the file system. A second run of `fsck` should be made to recheck the file system. If another allocated inode with a zero link count is found, the error message is repeated.
NO	Terminate the program.

Message

```
B BAD I=I
```

Inode *I* contains block number *B* with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system. This error condition may generate the `EXCESSIVE BAD BLKS` error message in Phase 1 if inode *I* has too many block numbers outside the file system range. This error condition generates the `BAD/DUP` error messages in Phases 2 and 4.

Message

```
EXCESSIVE BAD BLKS I=I (CONTINUE)
```

There are too many (usually more than 10) blocks with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system associated with inode *I*.

Possible responses to the `CONTINUE` prompt are:

YES	Ignore the rest of the blocks in this inode and continue checking with the next inode in the file system. This error condition prevents a complete check of the file system. A second run of `fsck` should be made to recheck this file system.
NO	Terminate the program.

Message

```
BAD STATE DDD TO BLKERR
```

An internal error has scrambled `fsck`'s state map to have the impossible value *DDD*. `fsck` exits immediately. If this occurs, contact your service representative or another qualified person.

Message

```
B DUP I=I
```

Inode *I* contains block number *B* that is already claimed by another inode. This error condition may generate the `EXCESSIVE DUP BLKS` error message in Phase 1 if inode *I* has too many block numbers claimed by other inodes. This error condition invokes Phase 1B and generates the `BAD/DUP` error message in Phases 2 and 4.

Message

```
BAD MODE: MAKE IT A FILE?
```

This message is generated when the status of a given inode is set to all ones, indicating file system damage. This message does not indicate disk damage, unless it appears repeatedly after `fsck -y` has been run. A response of `y` causes `fsck` to reinitialize the inode to a reasonable value.

Message

```
EXCESSIVE DUP BLKS I=I (CONTINUE)
```

There are too many (usually more than 10) blocks claimed by other inodes.

Possible responses to the `CONTINUE` prompt are:

YES	Ignore the rest of the blocks in this inode and continue checking with the next inode in the file system. This error condition

	prevents a complete check of the file system. A second run of `fsck` should be made to recheck the file system.
NO	Terminate the program.

Message

```
DUP TABLE OVERFLOW (CONTINUE)
```

An internal table in `fsck` containing duplicate block numbers has no more room.

Possible responses to the `CONTINUE` prompt are:

YES	Continue with the program. This error condition prevents a complete check of the file system. A second run of `fsck` should be made to recheck the file system. If another duplicate block is found, this error message will repeat.
NO	Terminate the program.

Message

```
PARTIALLY ALLOCATED INODE I=I (CLEAR)
```

Inode *I* is neither allocated nor unallocated.

Possible responses to the `CLEAR` prompt are:

YES	De-allocate inode *I* by zeroing out its contents.
NO	Ignore this error condition.

Message

```
INCORRECT BLOCK COUNT I=I (X should be Y) (CORRECT)
```

The block count for inode *I* is *X* blocks, but should be *Y* blocks. When preening, the count is corrected.

Possible responses to the CORRECT prompt are:

YES	Replace the block count of inode *I* by *Y*.
NO	Ignore this error condition.

Phase 1B: Rescan for More DUPS

When a duplicate block is found in the file system, the file system is rescanned to find the inode that previously claimed that block. When the duplicate block is found, the following informational message appears:

Message

```
B DUP I=I
```

Inode *I* contains block number *B* that is already claimed by another inode. This error condition generates the BAD/DUP error message in Phase 2. You can determine which inodes have overlapping blocks by examining this error condition and the DUP error condition in Phase 1.

Phase 2: Check Pathnames

This phase removes directory entries pointing to bad inodes found in Phases 1 and 1B. It reports error conditions resulting from:

- incorrect root inode mode and status
- directory inode pointers out of range
- directory entries pointing to bad inodes
- directory integrity checks

All errors in this phase are fatal if the file system is being preened, except for directories not being a multiple of the block size and extraneous hard links.

Phase 2 Error Messages

Message

```
ROOT INODE UNALLOCATED (ALLOCATE)
```

The root inode (usually inode number 2) has no allocate mode bits. This should never happen.

Possible responses to the `ALLOCATE` prompt are:

YES	Allocate inode 2 as the root inode. The files and directories usually found in the root will be recovered in Phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` will exit with the message

```
CANNOT ALLOCATE ROOT INODE
```

NO	Terminate the program.

Message

```
ROOT INODE NOT DIRECTORY (REALLOCATE)
```

The root inode (usually inode number 2) of the file system is not a directory inode.

Possible responses to the `REALLOCATE` prompt are:

YES	Clear the existing contents of the root inode and reallocate it. The files and directories usually found in the root will be recovered in Phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` will exit with the message:

```
CANNOT ALLOCATE ROOT INODE
```

NO	`fsck` will then prompt with `FIX`

Possible responses to the `FIX` prompt are:

YES	Change the type of the root inode to directory. If the root inode's data blocks are not directory blocks, many error messages will be generated.
NO	Terminate the program.

Message

```
DUPS/BAD IN ROOT INODE (REALLOCATE)
```

Phase 1 or Phase 1B has found duplicate blocks or bad blocks in the root inode (usually inode number 2) of the file system.

Possible responses to the `REALLOCATE` prompt are:

YES	Clear the existing contents of the root inode and reallocate it. The files and directories usually found in the root will be recovered in Phase 3 and put into the `lost+found` directory. If the attempt to allocate the root fails, `fsck` will exit with the message:

```
CANNOT ALLOCATE ROOT INODE
```

NO	`fsck` will then prompt with `CONTINUE`.

Possible responses to the `CONTINUE` prompt are:

YES	Ignore the `DUPS/BAD` error condition in the root inode and try to continue running the file system check. If the root inode is not correct, this may generate many other error messages.
NO	Terminate the program.

Message

```
NAME TOO LONG F
```

An excessively long path name has been found. This usually indicates loops in the file system name space. This can occur if a privileged user has made circular links to directories. These links must be removed.

Message

```
I OUT OF RANGE I=I NAME=F (REMOVE)
```

A directory entry *F* has an inode number *I* that is greater than the end of the inode list.

Possible responses to the REMOVE prompt are:

YES	Remove the directory entry *F*.
NO	Ignore this error condition.

Message

```
UNALLOCATED I=I OWNER=O MODE=M SIZE=S MTIME=T TYPE=F (REMOVE)
```

A directory or file entry *F* points to an unallocated inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and name *F* are printed.

Possible responses to the REMOVE prompt are:

YES	Remove the directory entry *F*.
NO	Ignore this error condition.

Message

```
DUP/BAD I=I OWNER=O MODE=M SIZE=S MTIME=T TYPE=F (REMOVE)
```

Phase 1 or Phase 1B has found duplicate blocks or bad blocks associated with directory or file entry *F*, inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the REMOVE prompt are:

YES	Remove the directory entry *F*.
NO	Ignore this error condition.

Message

```
ZERO LENGTH DIRECTORY I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(REMOVE)
```

A directory entry *F* has a size *S* that is zero. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the REMOVE prompt are:

YES	Remove the directory entry *F*; this will generate the BAD/DUP error message in Phase 4.
NO	Ignore this error condition.

Message

```
DIRECTORY TOO SHORT I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (FIX)
```

A directory *F* has been found whose size *S* is less than the minimum size directory. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the FIX prompt are:

YES	Increase the size of the directory to the minimum directory size.
NO	Ignore this directory.

Message

```
DIRECTORY F LENGTH S NOT MULTIPLE OF B (ADJUST)
```

A directory *F* has been found with size *S* that is not a multiple of the directory block size *B*.

Possible responses to the `ADJUST` prompt are:

YES	Round up the length to the appropriate block size. When preening the file system only a warning is printed and the directory is adjusted.
NO	Ignore the error condition.

Message

```
DIRECTORY CORRUPTED I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(SALVAGE)
```

A directory with an inconsistent internal state has been found.

Possible responses to the `SALVAGE` prompt are:

YES	Throw away all entries up to the next directory boundary (usually a 512-byte boundary). This drastic action can throw away up to 42 entries, and should be taken only after other recovery efforts have failed.
NO	Skip to the next directory boundary and resume reading, but do not modify the directory.

Message

```
BAD INODE NUMBER FOR '.' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(FIX)
```

A directory *I* has been found whose inode number for "." does not equal *I*.

Possible responses to the `FIX` prompt are:

YES	Change the inode number for "." to be equal to *I*.
NO	Leave the inode number for "." unchanged.

Message

```
MISSING '.' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (FIX)
```

A directory *I* has been found whose first entry is unallocated.

Possible responses to the `FIX` prompt are:

YES	Build an entry for "." with inode number equal to *I*.
NO	Leave the directory unchanged.

Message

```
MISSING '.' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, FIRST ENTRY IN DIRECTORY CONTAINS F
```

A directory *I* has been found whose first entry is *F*. `fsck` cannot resolve this problem. The file system should be mounted and entry *F* moved elsewhere. The file system should then be unmounted and `fsck` should be run again.

Message

```
MISSING '.' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, INSUFFICIENT SPACE TO ADD '.'
```

A directory *I* has been found whose first entry is not ".". This should never happen. `fsck` cannot resolve the problem. If this occurs, contact your service representative or another qualified person.

Message

```
EXTRA '.' ENTRY I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (FIX)
```

A directory *I* has been found that has more than one entry for ".".

Possible responses to the `FIX` prompt are:

YES	Remove the extra entry for ".".
NO	Leave the directory unchanged.

Message

```
BAD INODE NUMBER FOR '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(FIX)
```

A directory *I* has been found whose inode number for ".." does not equal the parent of *I*.

Possible responses to the `FIX` prompt are:

YES	Change the inode number for ".." to be equal to the parent of *I*. (Note that ".." in the root inode points to itself.)

NO Leave the inode number for ".." unchanged.

Message

```
MISSING '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (FIX)
```

A directory *I* has been found whose second entry is unallocated.

Possible responses to the `FIX` prompt are:

YES	Build an entry for ".." with inode number equal to the parent of *I*. (Note that ".." in the root inode points to itself.)
NO	Leave the directory unchanged.

Message

```
MISSING '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, SECOND ENTRY IN DIRECTORY CONTAINS F
```

A directory *I* has been found whose second entry is *F*. `fsck` cannot resolve this problem. The file system should be mounted and entry *F* moved elsewhere. The file system should then be unmounted and `fsck` should be run again.

Message

```
MISSING '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, INSUFFICIENT SPACE TO ADD '..'
```

A directory *I* has been found whose second entry is not "`..`" (the parent directory). `fsck` cannot resolve this problem. The file system should be mounted and the second entry in the directory moved elsewhere. The file system should then be unmounted and `fsck` should be run again.

Message

```
EXTRA '..' ENTRY I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F (FIX)
```

A directory *I* has been found that has more than one entry for ".." (the parent directory).

Possible responses to the `FIX` prompt are:

YES	Remove the extra entry for '..' (the parent directory).
NO	Leave the directory unchanged.

Message

```
N IS AN EXTRANEOUS HARD LINK TO A DIRECTORY D (REMOVE)
```

`fsck` has found a hard link *N* to a directory *D*. When preening the extraneous links are ignored.

Possible responses to the `REMOVE` prompt are:

YES	Delete the extraneous entry *N*.
NO	Ignore the error condition.

Message

```
BAD INODE S TO DESCEND
```

An internal error has caused an impossible state *S* to be passed to the routine that descends the file system directory structure. `fsck` exits. If this occurs, contact your service representative or another qualified person.

Message

```
BAD RETURN STATE S FROM DESCEND
```

An internal error has caused an impossible state *S* to be returned from the routine that descends the file system directory structure. `fsck` exits. If you encounter this error, contact your service representative or another qualified person.

Message

```
BAD STATE S FOR ROOT INODE
```

An internal error has caused an impossible state *S* to be assigned to the root inode. `fsck` exits. If this occurs, contact your service representative or another qualified person.

Phase 3: Check Connectivity

This phase checks the directories examined in Phase 2. It reports error conditions resulting from:

- unreferenced directories
- missing or full `lost+found` directories

Phase 3 Error Messages

Message

```
UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)
```

The directory inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of directory inode *I* are printed. When preening, the directory is reconnected if its size is non-zero; otherwise it is cleared.

Possible responses to the `RECONNECT` prompt are:

YES	Reconnect directory inode *I* to the file system in the directory for lost files (usually the `lost+found` directory). This may generate the `lost+found` error messages in Phase 3 if there are problems connecting directory inode *I* to the `lost+found` directory. It may also generate the `CONNECTED` error message in Phase 3 if the link was successful.
NO	Ignore this error condition. This generates the `UNREF` error message in Phase 4.

Message

```
NO lost+found DIRECTORY (CREATE)
```

There is no `lost+found` directory in the root directory of the file system; When preening `fsck` tries to create a `lost+found` directory.

Possible responses to the `CREATE` prompt are:

YES	Create a `lost+found` directory in the root of the file system. This may produce the message:

```
NO SPACE LEFT IN / (EXPAND)
```

See below for the possible responses. Inability to create a `lost+found` directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO	Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

The entry for `lost+found` is not a directory.

Possible responses to the REALLOCATE prompt are:

YES	Allocate a directory inode, and change `lost+found` to reference it. The previous inode referenced by the `lost+found` directory is not cleared. Thus it will either be reclaimed as an UNREF'ed inode or have its link count ADJUST'ed later in this phase. Inability to create a `lost+found` directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO	Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
NO SPACE LEFT IN /lost+found (EXPAND)
```

There is no space to add another entry to the `lost+found` directory in the root directory of the file system. When preening the `lost+found` directory is expanded.

Possible responses to the `EXPAND` prompt are:

YES
: Expand the `lost+found` directory to make room for the new entry. If the attempted expansion fails `fsck` prints the message:

```
SORRY. NO SPACE IN lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the `UNREF` error message in Phase 4. Clear out unnecessary entries in the `lost+found` directory. This error is fatal if the file system is being preened.

NO
: Abort the attempt to link up the lost inode. This generates the `UNREF` error message in Phase 4.

Message

```
DIR I=I1 CONNECTED. PARENT WAS I=I2
```

This is an advisory message indicating that a directory inode *I1* was successfully connected to the `lost+found` directory. The parent inode *I2* of the directory inode *I1* is replaced by the inode number of the `lost+found` directory.

Message

```
DIRECTORY F LENGTH S NOT MULTIPLE OF B (ADJUST)
```

A directory *F* has been found with size *S* that is not a multiple of the directory block size *B*. (Note that this may reoccur in Phase 3 if the error condition is not corrected in Phase 2).

Possible responses to the `ADJUST` prompt are:

YES	Round up the length to the appropriate block size. When preening the file system only a warning is printed and the directory is adjusted.
NO	Ignore the error condition.

Message

```
BAD INODE S TO DESCEND
```

An internal error has caused an impossible state *S* to be passed to the routine that descends the file system directory structure. `fsck` exits. If this occurs, contact your service representative or another qualified person.

Phase 4: Check Reference Counts

This phase checks the link count information obtained in Phases 2 and 3. It reports error conditions resulting from:

- unreferenced files
- missing or full `lost+found` directory
- incorrect link counts for files, directories, symbolic links, or special files

- unreferenced files, symbolic links, and directories
- bad or duplicate blocks in files, symbolic links, and directories

All errors in this phase (except running out of space in the `lost+found` directory) are correctable if the file system is being preened.

Phase 4 Error Messages

Message

```
UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)
```

Inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. When preening the file is cleared if either its size or its link count is zero; otherwise it is reconnected.

Possible responses to the `RECONNECT` prompt are:

YES	Reconnect inode *I* to the file system in the directory for lost files (usually the `lost+found` directory). This may generate the `lost+found` error message in Phase 4 if there are problems connecting inode *I* to the `lost+found` directory.
NO	Ignore this error condition. This will always invoke the `CLEAR` error condition in Phase 4.

Message

```
(CLEAR)
```

The inode mentioned in the error message immediately preceding cannot be reconnected. This message cannot appear if the file system is being preened, because lack of space to reconnect files is a fatal error.

Possible responses to the CLEAR prompt are:

YES	De-allocate the inode by zeroing out its contents.
NO	Ignore this error condition.

Message

```
NO lost+found DIRECTORY (CREATE)
```

There is no lost+found directory in the root directory of the file system; when preening fsck tries to create a lost+found directory.

Possible responses to the CREATE prompt are:

YES Create a lost+found directory in the root of the file system. This may generate the message:

```
NO SPACE LEFT IN / (EXPAND)
```

See below for the possible responses. Inability to create a lost+found directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

The entry for `lost+found` is not a directory.

Possible responses to the `REALLOCATE` prompt are:

YES — Allocate a directory inode and change the `lost+found` directory to reference it. The previous inode reference by the `lost+found` directory is not cleared. Thus it will either be reclaimed as an `UNREF`'ed inode or have its link count `ADJUST`'ed later in this phase. Inability to create a `lost+found` directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This generates the `UNREF` error message in Phase 4.

NO — Abort the attempt to link up the lost inode. This generates the `UNREF` error message in Phase 4.

Message

```
NO SPACE LEFT IN /lost+found (EXPAND)
```

There is no space to add another entry to the `lost+found` directory in the root directory of the file system. When preening the `lost+found` directory is expanded.

Possible responses to the EXPAND prompt are:

YES Expand the lost+found directory to make room for the new entry. If the attempted expansion fails fsck prints the message:

```
SORRY. NO SPACE IN lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This generates the UNREF error message in Phase 4. Clear out unnecessary entries in the lost+found directory. This error is fatal if the file system is being preened.

NO Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
LINK COUNT TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X
SHOULD BE Y (ADJUST)
```

The link count for inode *I* is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* are printed. When preening the link count is adjusted unless the number of references is increasing, a condition that should never occur unless precipitated by a hardware failure. When the number of references is increasing during preening, fsck exits with the message:

```
LINK COUNT INCREASING
```

Possible responses to the ADJUST prompt are:

YES	Replace the link count of file inode *I* by *Y*.
NO	Ignore this error condition.

Message

```
UNREF TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
```

Inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. Since this is a file that was not connected because its size or link count was zero, it is cleared during preening.

Possible responses to the `CLEAR` prompt are:

YES	De-allocate inode *I* by zeroing out its contents.
NO	Ignore this error condition.

Message

```
BAD/DUP TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
```

Phase 1 or Phase 1B has found duplicate blocks or bad blocks associated with inode *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. This message cannot appear when the file system is being preened, because it would have caused a fatal error earlier.

Possible responses to the `CLEAR` prompt are:

YES	De-allocate inode *I* by zeroing out its contents.
NO	Ignore this error condition.

Phase 5: Check Cylinder Groups

This phase checks the free block and used inode maps. It reports error conditions resulting from:

- allocated blocks in the free block maps
- free blocks missing from free block maps
- incorrect total free block count
- free inodes in the used inode maps
- allocated inodes missing from used inode maps
- incorrect total used inode count

Phase 5 Error Messages

Message

```
CG C: BAD MAGIC NUMBER
```

The magic number of cylinder group *C* is wrong. This usually indicates that the cylinder group maps have been destroyed. When running interactively, the cylinder group is marked as needing reconstruction. This error is fatal if the file system is being preened.

Message

```
BLK(S) MISSING IN BIT MAPS (SALVAGE)
```

A cylinder group block map is missing some free blocks. During preening the maps are reconstructed.

Possible responses to the `SALVAGE` prompt are:

YES	Reconstruct the free block map.
NO	Ignore this error condition.

Message

```
SUMMARY INFORMATION BAD (SALVAGE)
```

The summary information was found to be incorrect. When preening, the summary information is recomputed.

Possible responses to the SALVAGE prompt are:

YES	Reconstruct the summary information.
NO	Ignore this error condition.

Message

```
FREE BLK COUNT(S) WRONG IN SUPERBLOCK (SALVAGE)
```

The super-block free block information was found to be incorrect. When preening, the super-block free block information is recomputed.

Possible responses to the SALVAGE prompt are:

YES	Reconstruct the super-block free block information.
NO	Ignore this error condition.

Cleanup Phase

After a file system has been checked, a few cleanup functions are performed. The cleanup phase displays advisory messages about the status of the file system.

Message

```
V files, W used, X free (Y frags, Z blocks)
```

This is an advisory message indicating that the file system checked contains *V* files using *W* fragment sized blocks, and that there are *X* fragment sized blocks free in the file system. The numbers in parentheses break the free count down into *Y* free fragments and *Z* free full sized blocks.

Message

```
***** FILE SYSTEM WAS MODIFIED *****
```

This is an advisory message indicating that the file system was modified by `fsck`. If this file system is mounted or is the current root file system, you should reboot. If the file system is mounted you may need to unmount it and run `fsck` again; otherwise the work done by `fsck` may be undone by the in-core copies of tables.

Checking `bfs` File Systems

All I/O for `bfs` file systems is synchronous; therefore, `bfs` file systems will not get corrupted even if proper shutdown procedures are not observed.

Corruption of a `bfs` file system is likely to occur only if the system crashes during the process of compaction. See the section "Compaction" earlier in this chapter for a description of compaction.

`fsck` checks the sanity words stored in the `bfs` super-block to see if compaction was in process before the system crashed. If it was, `fsck` completes the compaction of the file system.

6 Machine Management

Introduction

This chapter tells you how to perform tasks that affect the way in which your computer operates or that provide information on the current state of your computer.

Some of the tasks described in this chapter can be performed through the sysadm system administration interface. Others require execution of shell-level commands, and some require knowledge of firmware-level operations available on your computer. The operations described in this chapter, and the ways that you can perform them, are:

Figure 6-1: Machine Management Tasks

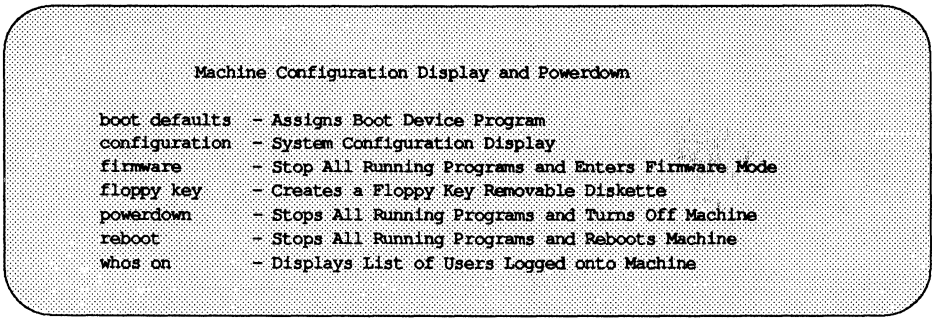
Task	Interface
Changing your boot default parameters	sysadm/shell
Changing to firmware mode	sysadm/shell
Creating a floppy key	sysadm/firmware
Powering down and rebooting your system	sysadm/shell
Displaying system configuration information	sysadm/shell
Displaying information about the users on your system	sysadm/shell
Returning from firmware	firmware
Making new bootable disks	shell

System Administration Interface

To access the system administration menu for machine management, log in as root and type:

```
sysadm machine
```

The following menu will appear on your screen:



```
                Machine Configuration Display and Powerdown

boot defaults   - Assigns Boot Device Program
configuration   - System Configuration Display
firmware        - Stop All Running Programs and Enters Firmware Mode
floppy key      - Creates a Floppy Key Removable Diskette
powerdown       - Stops All Running Programs and Turns Off Machine
reboot          - Stops All Running Programs and Reboots Machine
whos on         - Displays List of Users Logged onto Machine
```

If you prefer not to use the menus, you can perform the same tasks by executing shell-level commands instead. The following table shows the shell commands that correspond to the tasks listed on the menu.

Figure 6-2: Machine Tasks and Shell Commands

Task to Be Performed	`sysadm` Task	Shell Command
Setting manual boot defaults	`boot defaults`	`fltboot`(1M)
Changing to firmware mode	`firmware`	`shutdown`(1M)
Creating a floppy key	`floppy key`	`n/a`
Powering down your machine	`powerdown`	`shutdown`(1M)
Rebooting your system	`reboot`	`shutdown`(1M)
Displaying configuration info.	`summary`	`prtconf`(1)
Displaying system name	system	`uname`(1)
Displaying who is logged on	`whos on`	`who`(1)

Each task listed above is explained fully later in this chapter. Details on each command can be found in the *System Administrator's Reference Manual*.

The remainder of this chapter provides background information and describes how to perform all the tasks mentioned in Figure 6-1 through the shell or from firmware mode, as appropriate.

If you are a non-expert administrator, you are encouraged to perform these tasks, where possible, through the `sysadm` interface.

The next section provides some guidelines that should be followed whenever you perform machine management tasks.

An Overview of Machine Management

The task of managing a computer involves many responsibilities. Typically, the actions you take when performing machine management tasks will affect the operation of the computer as a whole, and every user on the computer.

In addition to the information in this chapter, you will also want to consult the hardware manual that accompanies your computer. This hardware manual will provide a detailed discussion of machine management and descriptions of firmware commands that are available on your computer.

Many administrative tasks require the system to be shut down to a system state other than multi-user state (see "System States" later in this chapter for a description).

In many system states, conventional users cannot access the system. Therefore, you should try to perform tasks that require a change in system state at at time when you will interfere least with users' work. Sometimes situations arise that require the system to be taken down with little or no warning to the users. Try to give users as much advance notice as possible about events affecting the use of the machine. When you must take the system out of service, tell them when it will become available again.

Use the news (see `news`(1)) and the message of the day (contained in the file `/etc/motd`) to keep users informed about changes in hardware, software, policies, and procedures.

Follow the guidelines below whenever you do a task that requires the system to leave multi-user state.

1. Schedule tasks that will affect service for periods of low system use. Inform users of the times that the system will be unavailable.
2. Check to see who is logged in before taking any actions that would affect active users. The `whodo`(1M) and `who`(1) commands can be used to see who is on the system.
3. If the system is in use, warn the users about changes in system states or pending maintenance actions. If you must interrupt service immediately, broadcast a warning to all users' screens with the `wall`(1M) command. Give users a reasonable amount of time to stop working and log off before you take the system down.

Keep a record of your administrative activities in a system log notebook (if you have one). This log can prove invaluable in recovering your system should you make a serious error.

The remainder of this chapter provides:

- a brief overview of the boot procedure that tells you how the operating system is loaded from disk into memory and executed
- an overview of system states that tells you what happens after the machine is booted and it enters the default system state, how to change the default system state, and how to change system states after powerup
- detailed procedures for the tasks listed in Figure 6-1

In previous releases of UNIX System V, it was assumed that a System V (`s5`) file system was defined on the root partition, and that the programs and data files needed during booting (and in firmware mode) reside in the root file system.

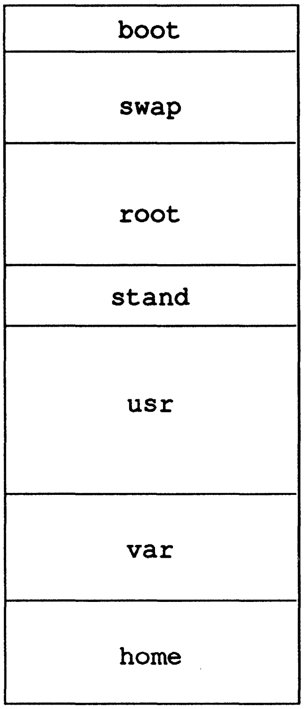
With UNIX System V Release 4.0, the boot procedure no longer depends on the file system type of the root file system. This file system independent boot procedure is implemented through the use of a new file system type, the boot file system type (`bfs`).

The following section explains the location and contents of the boot file system. It also explains the location and contents of the boot partition, a separate disk partition (not a file system) that holds the programs used to load the bootable programs found in the boot file system.

The stand and boot Partitions

Figure 6-3 shows the general layout of a single disk formatted into the default partitions. The disk layout shown in the figure is a generalized single-disk layout; the layout on your machine may be different. (See Appendix A for the default partitioning on the AT&T 3B2 Computer.)

Figure 6-3: Generalized Hard Disk Default Partitioning



The two partitions of interest to the boot procedure are the boot and stand partitions.

The boot partition is not a file system, but a special area of the disk that contains the boot programs. On most computers, the boot process is basically a three step process. The following is the process as executed on the AT&T 3B2 Computer:

- you turn your machine on, and the firmware loads the mboot program from the boot partition into a special area of memory

- `mboot` in turn loads another program from the boot partition, called `boot`
- `boot` then loads the bootable operating system `/stand/unix`

The bootable operating system `unix` is found in the `stand` partition. This partition has a file system defined on it that contains all the bootable programs and data files used during the boot procedure. It can be identified using the `prtvtoc`(1M) command as the partition with the tag of `6` (indicating `V_STAND`). The file system defined on this partition is mounted by default as `/stand`.

The contents of `/stand` on the AT&T 3B2 Computer include the following:

`unix`	This is the bootable operating system. When the boot program is loaded, it searches for and loads this program into memory, and then passes control to it. Once the bootable operating system is running, various system daemons are started, and the system enters one of the `init` states (see the description of `/sbin/inittab` in this chapter). Note that the filename `unix` is linked to `/stand/unix` for compatibility with earlier releases; you can enter `unix` or `/stand/unix` at any prompt requiring the name of the bootable operating system.
`system`	This is the system configuration file; it contains a description of the hardware and software modules that must be included in the bootable operating system (`/stand/unix`) for correct configuration of the system. If the time stamp of the `system` file is newer than the time stamp of the `/stand/unix` file, a reconfiguration of the bootable operating system is necessary before the boot program can pass control to it. Note that the filename `system` is linked to `/stand/system` for compatibility with earlier releases; you can enter `system` or `/stand/system` at any prompt requiring the name of the system configuration file.
`mUNIX`	This is a version of the UNIX operating system that is run only during the configuration of a new bootable operating system (`unix`).

mini_system	This is the system configuration file for the mUNIX program.
filledt	This is a bootable program that completes the equipped device table for the machine; it is run from firmware mode only. It is described in the hardware documentation that accompanies the AT&T 3B2 Computer.
dgmon	This is a firmware command that runs the diagnostics for the computer; it is run from firmware mode only. It is described in the hardware documentation that accompanies the AT&T 3B2 Computer.

See "Configuring the UNIX Operating System" in the "Performance Management" chapter for more information on the operating system files in /stand.

Operations on the boot Partition

The only operation performed on the boot partition is the loading of the boot programs; no file system is defined on this partition.

The boot programs are loaded using the newboot(1M) command. The boot programs are found in the root file system under the /usr/lib directory. The command is used as follows:

```
# newboot /usr/lib/boot /usr/lib/mboot /dev/rdsk/c1d1s7
```

The special file used in the example above may vary depending on the number of disks you have connected to your computer. See newboot(1M) for a complete description of this command.

This command is normally not used unless you are manually repartitioning your hard disks or creating a new bootable disk (see "Making New Bootable Disks," in this chapter). In the delivered system, a boot partition is defined on the first integral hard disk. This partition already contains the boot and mboot programs.

Operations on the stand Partition

In the delivered system, the `stand` partition contains a file system of type `bfs`; this boot file system is mounted as `/stand` by default.

The boot file system is a flat file system, with only one directory. You can copy and move regular files to and from a boot file system, but cannot make directories or create other special files in the `bfs` file system. You can, however, use file system commands such as `mount`(1M), `umount`(1M), and `fsck`(1M).

It is recommended that you use this file system for boot- and configuration-related files only.

If you want to, you can use `mkfs` to create `bfs`-type file systems on other disk partitions or on other disks (to make them bootable). See the section in this chapter "Making New Bootable Disks," the "File System Administration" chapter, and the `mkfs`(1M) manual page for more information. Having multiple boot file systems on one or more disks is particularly useful in an operating system development environment.

CAUTION You must define `bfs`-type file systems only on hard disk partitions with a `tag` of `6`. Do not make `bfs`-type file systems on any other devices. Similarly, do not make a file system of any other type than `bfs` in a partition with a `tag` of `6`. Doing so may render your machine unbootable. See the references cited in the above paragraph, and the `fmthard`(1M) and `prtvtoc`(1M) manual pages in the *System Administrator's Reference Manual* for more information.

Any disk can contain `stand` and `boot` partitions. Thus it is possible to boot the system from any disk that has both, as long as the root file system is also accessible (though not necessarily on the same disk).

A disk can also have multiple `stand` partitions with a boot file system defined on each one. Each partition can contain a version of the bootable programs. The `next part` firmware command (described below) can be used to switch partitions from which you can request a manual boot from firmware mode. You can manually request a boot from any `stand` partition on any disk from the firmware prompt.

CAUTION Note that while you can request a boot from any `stand` partition on any disk from firmware, configuration of a new operating system will work only on the `stand` partition found in `/etc/vfstab`. For example, if you specify a `system` file at the firmware prompt that is in a `stand` file system other than the one found in `/etc/vfstab`, the configuration process will fail, even if you made the correct `/stand` partition the current partition using the `next part` command.

Although any disk can be made bootable, only one disk can be booted on powerup, and that is the default boot disk defined in Non-Volatile RAM (NVRAM). You can change the default boot disk through the procedure in this chapter titled "Changing Default Boot Parameters." If you define multiple `stand` partitions on the default boot disk, the one used during a reboot or powerup will be the one with the smallest partition number.

For example, suppose you have a default boot disk that has 3 `stand` partitions, all with `bfs`-type file systems defined on them that contain one or more bootable programs (such as `unix` or `dgmon`), and the device names for the three partitions are `/dev/dsk/c1d0s3` for slice 3, `/dev/dsk/c1d0s5` for slice 5, and `/dev/dsk/c1d0s8` for slice 8. On reboot and powerup, the system will boot from slice 3 (`/dev/dsk/c1d0s3`); the other `stand` partitions will be ignored.

When you are in firmware mode, however, you can access all `stand` partitions on a disk. When you enter firmware mode, the following prompt is displayed:

```
        Enter name of program to execute [ ]:
```

If the brackets are empty, as shown above, just press RETURN. The following prompt is displayed, after a list of possible load devices:

```
        Enter Load Device Option Number [1 (HD70)]:
```

After pressing RETURN at this prompt (regardless of whether you change the load device or accept the default shown in brackets), the following prompt is displayed:

```
        Enter name of boot file (or 'q' to quit):
```

Pressing RETURN at the above prompt lists the contents of the first boot partition, `/dev/dsk/c1d0s3`. This partition is the working partition.

If you enter `next part` at the prompt, slice 5 (`/dev/dsk/c1d0s5`) becomes the working partition. You can then either choose a program from this partition or enter `next part` again. If you enter `next part`, slice 8 (`/dev/dsk/c1d0s8`) becomes the working partition.

Once you reach the last `stand` partition defined on the disk, you can enter `next part` to make the first boot file system the working partition again (in our example, slice 3 (`/dev/dsk/c1d0s3`)).

See the section, "Changing Default Boot Parameters", in this chapter for more detail on firmware prompts and commands.

The file `/etc/vfstab` contains the pathname of the `stand` partition to be mounted during single- and multi-user states and is used during configuration of a new operating system on powerup or reboot. This should always be the one defined on the lowest numbered partition on your default boot disk (usually partition 3). The device name has the form `/dev/dsk/c?d?s3`.

Boot Scenarios

There are several reasons that a system boot takes place:

- Power is newly applied to the machine. In this case, the boot program looks for the `unix` and `system` files in the first boot partition on the default boot disk. If `unix` has the same or a later timestamp than `system`, then the boot program loads and executes `unix`; if `system` is newer, the boot program will execute `mUNIX`, and then `buildsys`(1M) to configure a new `unix`. When the new `unix` is built, the system boots the new `unix` and the system comes up in the state defined by the `initdefault` entry in `/sbin/inittab`. (See `inittab`(4).)
- A reboot of the machine is explicitly requested while the UNIX system is running (e.g., `init 6` or `shutdown -i6`). In this case, all system activity is stopped (all processes are killed, etc.). The system checks to see if a new `unix` needs to be configured (as above), and if so, configures one. Then, the system unmounts all file systems and boots `unix`. The system comes up in the state defined by the `initdefault` entry in `/sbin/inittab`.
- A reboot is requested from firmware. From the firmware prompt, the user can specify the name of any executable (such as `unix`) or text file (such as `system`) in `/stand`. If an executable is specified, the boot program loads it and branches to it. If a text file is specified, the boot

program starts `mUNIX`, and then executes `buildsys`(1M) to configure a new `unix`, using the text file specified as the `system` file for the new `unix`. When the new `unix` is built, the system loads and executes it, and comes up in the state defined by the `initdefault` entry in `/sbin/inittab`.

- A crash occurs and the machine automatically reboots. In this case, the procedure is the same as for the first case, above.

See the section, "Configuring the UNIX Operating System," in the "Performance Management" chapter for a description of the configuration process.

The /boot Directory

There is a directory called `/boot` in the root file system that is not to be confused with the `stand` and `boot` partitions. Files in the `/boot` directory are used during the configuration of a new `unix`. See "Configuring the UNIX Operating System," in the "Performance Management" chapter, for more details.

System States

Once powered up, the UNIX system operates in one system state or another. Having different system states allows you to perform administrative and operational functions with the confidence that the computer is operating with a selected group of active processes.

There are many system states, but first a note of clarification. Many terms are used to identify the particular operating state of the system. The system state is also called the "init state," "run state," "run level," or "run mode." In this guide, the term system state is used to identify the system's operating state.

There are several ways that you can change from one system state to another:

- with the `shutdown` command
- with the `powerdown` command
- with the `init` command

The first two commands call the `init` command during their execution. (Refer to the *System Administrator's Reference Manual* for complete information on using and changing system states with these three commands.) Figure 6-4 shows the factory-configured states in which your system can operate.

Figure 6-4: System States

System State	Description
0	Powerdown state. Shut the machine down so it is safe to remove the power. Have the machine remove its own power (if possible).
1	State 1 is referred to as the administrative state. In state 1 file systems required for multi-user operations are mounted, and logins requiring access to multi-user file systems can be used. Other multi-user services are unavailable. Note that going to state 1 is only meaningful when the system is coming up from firmware or state s(S). When going to state 1 from state 2, no services are stopped, no processes are killed, and the system continues operating just as in state 2.
s, or S	State s (or S) is referred to as the single-user state. It is used to install or remove software utilities, run file system backups or restores, and check file systems. All multi-user file systems are unmounted and the system can only be accessed through the console. Logins requiring access to multi-user file systems cannot be used.
2	The normal operating state for the system (also called multi-user state) as delivered. File systems are mounted, and multi-user services are started.
3	Used to start Remote File Sharing (RFS), connect your computer to an RFS network, mount remote resources, and offer your resources automatically. Known as the RFS state.
4	User-defined system state. This state is not used in the delivered system.
5	Firmware state. This state is used to run special firmware commands and programs that reside in Non-Volatile Random Access Memory (NVRAM), and to run programs that reside in the `/stand` file system under the control of the NVRAM. An example of the former is making a floppy key (see "Making a Floppy Key" in the "System Setup" chapter). An example of the latter is executing `/stand/unix` to reboot the system.

Figure 6-4: System States (continued)

6	Halt and reboot the operating system to the state defined by the `initdefault` entry in the `/sbin/inittab` file.
a, b, or c	Process those `/sbin/inittab` file entries assigned the *a, b,* or *c* system state. These are pseudo-states that may be used to run certain commands without changing the current system state. They are supported by `init` but unused in the delivered system.
Q or q	Re-examine the `/sbin/inittab` file.

As delivered, the system enters the multi-user state on powerup. File systems are mounted, daemons started, and system services (such as `lp` and `uucp`) are made available. These activities are performed by the `rc2` script (see `rc2`(1M)).

Not all activities, however, should be performed in multi-user state. For example, you should never change your system's configuration while users are accessing it because much data may be lost. By changing the system to single-user state, you can be sure that you are the only person logged on to the system and, therefore, that it is safe to perform critical system administration tasks, such as changing the system configuration.

Entering the Multi-User State During Power Up

When you power up or reboot your system, it enters the default system state that appears in the `initdefault` line of the `/sbin/inittab` file. Normally, this line contains a `2` in the second field, indicating that the system is to be put into multi-user state by default. The following is an example `initdefault` entry:

```
    is:2:initdefault:
```

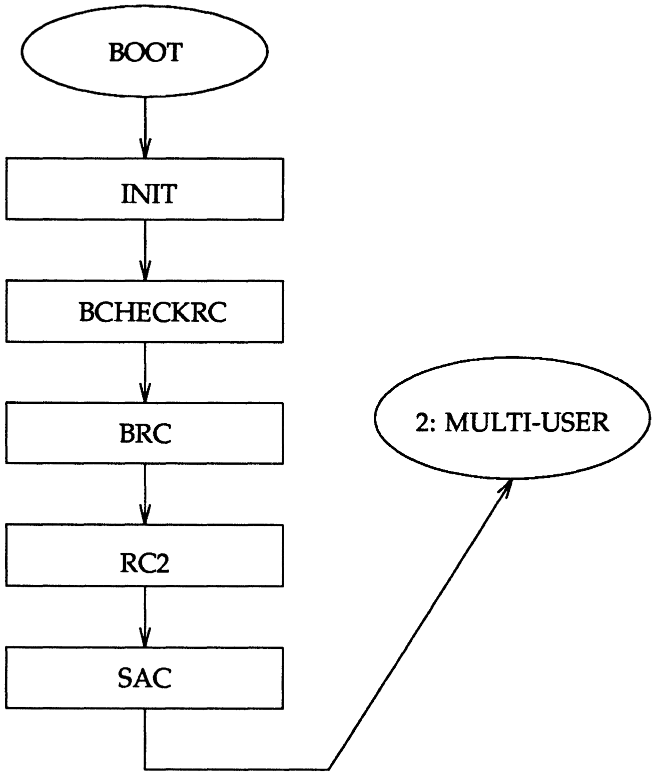
You can change the default run state of the computer by changing the second field of the `initdefault` line in your `/sbin/inittab` file. However, do not change the second field to a `1` or an `S` as you may render the system unbootable. Use a lowercase `s` when you want the default system state to be single-user.

Once the machine boots and control passes to the `unix` program, the following events occur as the system goes to multi-user state:

1. Early system initializations are started by `init`.
2. File system are checked and mounted by `bcheckrc`.
3. Other startup function are performed by `brc`.
4. The system state change is prepared by the `rc2` procedure.
5. The system is made public via the spawning of the service access facility (`ttymon` and `sac`).

Figure 6-5 shows the most important events that occur as `unix` comes up in multi-user state.

Figure 6-5: A Look at System Initialization



Early Initialization

After the operating system is loaded into core memory via the boot programs (see the section "The Boot Procedure," in this chapter) the `init` process is created. It immediately scans `/sbin/inittab` for entries of the type `sysinit`. A sample listing of these entries is shown in Figure 6-6.

Figure 6-6: Sample `sysinit` Entries in an `/sbin/inittab` File

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap
fs::sysinit:/sbin/bcheckrc </dev/console >/dev/console 2>&1
ck::sysinit:/sbin/setclk </dev/console >/dev/console 2>&1
xdc::sysinit:/sbin/sh -c 'if [ -x /etc/rc.d/0xdc ] ; then /etc/rc.d/0xdc ;
fi' >/dev/console 2>&1
ac::sysinit:/sbin/ckmunix </dev/console >/dev/console 2>&1
```

These entries are executed in the order in which they are listed in the file. Once the first entry is executed, communication between the system and the console is established. Subsequent entries define the standard input and the standard output as `/dev/console`.

Preparing the System State Change

Now the system must be placed in a particular system state. First, `init` scans each entry in the `/sbin/inittab` file for the value `initdefault` in the third field (the "action" field). Including the value `initdefault` in the third field of an entry means that the default system state is defined in the second field of that entry. For example, in the first line of Figure 6-7, the 2 in the second field, followed by the value `initdefault` in the third field, means that the default system state for this system is system state 2 (multi-user state).

Once `init` has identified the default system state as system state 2, it searches the table for all entries that specify processes for system state 2. Typical system state 2 entries are shown in Figure 6-7.

Figure 6-7: System State 2 Processes

```
s2:23:wait:/sbin/rc2 >/dev/console 2>&1 </dev/console
sc:234:respawn:/usr/lib/saf/sac -t 300
co:1234:respawn:/usr/lib/saf/ttymon -g -p "Console Login: " -m ldterm
-d /dev/console -l console
ct:234:respawn:/usr/lib/saf/ttymon -g -m ldterm -d /dev/contty -l
contty
he:234:respawn:/usr/sbin/hdelogger
```

The processes defined in these entries are executed before the system enters multi-user system state during powerup (see "System State Directories" later in this chapter). The `rc2` script accomplishes (among other things) the following:

- set up and mount the file systems
- start the `cron` daemon
- display the current system hardware configuration
- make uucp available for use (if installed)
- make the LP print service available for use (if installed)

Next, `init` searches the `/sbin/inittab` file for system state 2 processes and executes them in the order found in the file. In this example, `init` performs the following functions:

- starts the service access controller (`sac`) for the ports
- starts the tty monitor (`ttymon`) for the console
- starts the error logging daemon (`hdelogger`) for the hard disk

When this is complete, the full multi-user environment is established and the system is in system state 2. The system is now available for users to log in as shown by the `login:` prompt that appears on the user's terminals.

Changing System States After Power Up

There may be times when you want to change system states after you have powered up your computer. For example, you may want to perform system diagnostics, which require you to have your computer in the firmware state (system state 5).

For most administrative duties, you will want to change the system state from multi-user to system states `s, 1, or 5`. The general procedure used to change system states once the computer is powered up is as follows:

1. Log in as `root`. You will be prompted for the root password. After you have entered it, the root prompt (#) will appear.
2. Check to see if any users are on the system by typing:

   ```
   who -H
   ```

   A typical response might be:

   ```
   NAME      LINE       TIME
   root      console    Aug 31 10:28
   marcus    term/12    Aug 31 09:43
   ```

   If there are any users logged in to the system, go to step 3; if not, go to step 4.
3. To notify users that the system is about to be shut down, type `wall`, followed (on one or more separate lines) by a message announcing the shutdown. To end your message, press RETURN and then type CTRL-d, as shown in the following example:

```
$ wall
The system will be coming down in 5 minutes.
Please log off.
CTRL-d


Broadcast Message from root (console) on unix Wed Oct 5 07:30:27...
The system will be coming down in 5 minutes.
Please log off.
```

4. The shutdown or init command is executed with an argument that specifies the desired system state. (See Figure 6-4 for a list of available arguments.)

   The init process searches the /sbin/inittab file for processes that match the new system state and executes them in the order that they appear in the file.

   Key procedures (such as /sbin/shutdown, /sbin/rc0, /sbin/rc2, and /sbin/rc3) are run to initialize the new state.

   The system then enters the new state. If the new system state is either state s (single-user) or state 5 (firmware), sensitive administrative procedures can then be performed.

Changing to Single-User State (System State s)

Some administrative functions, such as backing up the hard disk, can be done only when the system is in the single-user state. The normal way to go to single-user state is by running the shutdown command. This command executes all the files in the /sbin/rc0.d and /sbin/shutdown.d directories by invoking the /sbin/rc0 procedure. The shutdown command accomplishes, among other things, the following:

- closes all open files and stops all user processes

- stops all daemons and services
- writes all system buffers out to the disk
- unmounts all file systems necessary for multi-user operations, but not needed in single-user state (such as /home)

There are two ways to change to single-user state:

1. you can run the `shutdown -is` command (recommended)
2. you can run the `init s` command

After these programs complete the change to single-user state, you get the message:

```
INIT: SINGLE USER MODE

Type Ctrl-d to proceed with normal startup
(or give root password for system maintenance:
```

Type the root password, press RETURN and you will get the single-user prompt (#). You are now ready to perform tasks that should be done only in the single-user state.

Changing to Multi-User State (System State 2)

System state 2 is the normal operating state of the system when it is not connected to a network. In this system state, several users can be logged in at once and use the system's resources. To change to multi-user state, run:

```
init 2
```

This procedure executes the /sbin/rc2 script which runs processes in the /sbin/rc2.d directory. Running `init 2` also initiates the Service Access Facility which manages access to your system through ports and other communication devices. (See the "Service Access" chapter for details.)

CAUTION The /var file system must be mounted before executing `init 2`.

Changing to RFS State (System State 3)

Before you can perform administrative tasks associated with the Remote File Sharing (RFS) utilities, you must change the system to system state 3 (defined as the RFS state). To change to the RFS state, run:

```
init 3
```

This procedure executes both the /sbin/rc2 and /sbin/rc3 scripts. These scripts run processes in all directories associated with system states 2 and 3, respectively. Running `init 3` also initiates the Service Access Facility which manages access to your system through ports and other communication devices. (See the "Service Access" chapter for details.)

In addition to the scripts in /sbin/rc2.d (multi-user state directory), scripts in /sbin/rc3.d (the RFS state directory) are run to do the following, if configured:

- start Remote File Sharing and connect your computer to a Remote File Sharing network
- advertise your resources to remote computers
- mount remote resources on your computer

If you are in a Remote File Sharing environment, you may want to change your `initdefault` entry in /sbin/inittab from 2 to 3, so that you automatically come up in Remote File Sharing state when you reboot your computer.

CAUTION The /var file system must be mounted before executing `init 3`.

Changing to Firmware State and Reboot States (System States 5 and 6)

Firmware state (defined as system state 5) is used both to access programs that reside in Non-Volatile Random Access Memory (NVRAM) and to run programs in the root file system under the control of the NVRAM. An example of the former is making a floppy key. An example of the latter is executing `/stand/unix` to reboot the system.

Rebooting the operating system (defined as system state 6) kills most active processes and forces the machine into a condition similar to that during powerup. After you install a software package that requires reinitialization of the system for proper operation, reboot the computer.

To go to the firmware state or to reboot your computer after reconfiguring it, run the `shutdown` command with the `-i5` or the `-i6` option. System states 5 and 6 are similar because both can exist only when the computer is under firmware control. The difference between the two states is one of duration. Your computer will stay in state 5 as long as you desire; it will stay in state 6 only long enough to reconfigure the operating system (if necessary) and to restart the initialization of it (that is, to reboot the system).

Figure 6-8: System State 5 and 6 Processes (from the Sample `/sbin/inittab` File)

```
f1:056:wait:/sbin/led -f     # start green LED flashing
s0:5:wait:/sbin/rc0 firmware >/dev/console 2>&1 </dev/console
s6:6:wait:/sbin/rc6 reboot >/dev/console 2>&1 < /dev/console
```

The `rc0` script runs processes in all directories associated with system state `0`. The `rc6` script runs processes in all directories associated with system state `6`.

Turning the System Off

To turn off your system, use the `shutdown -i0` command. (There may also be a switch on the cabinet of your computer that allows you to power down the machine. See the installation guide for your computer.) The following entries in the `/sbin/inittab` file apply to powering the system down:

Figure 6-9: System State 0 Processes (from the Sample `/sbin/inittab` File)

```
fl:056:wait:/sbin/led -f     # start green LED flashing
s0:0:wait:/sbin/rc0 off >/dev/console 2>&1 </dev/console
```

The `rc0` procedure is called to clean up and stop all user processes, daemons, and other services, to unmount the file systems and remove power from the machine.

System State Directories

System states 0, 2, and 3 each have a directory of files that are executed in transitions to and from that state. These directories are `/sbin/rc0.d`, `/sbin/rc2.d`, and `/sbin/rc3.d`, respectively. Most files in these directories are linked to files in the `/sbin/init.d` directory. Typically, their purpose is to start and stop various system services or daemons.

The system state files are named according to the following conventions:

S*NNname*

or

K*NNname*

Each file name consists of three parts:

S or K
: The first letter specifies whether the process should be started (S) or killed (K) on entering the new system state.

NN
: The next two characters are a number from 00 to 99. They show the order in which the files will be started (S00, S01, S02, and so on) or killed (K00, K01, K02, and so on).

name
: The rest of the file name is the name of the file in the `/sbin/init.d` directory to which this file is linked.

For example, the `/sbin/init.d/cron` shell script is linked to the `/etc/rc2.d/S75cron` and `/etc/rc0.d/K70cron` files. The `/sbin/init.d/cron` script will execute `/usr/bin/cron` when run with the `start` option; it will kill the `cron` process when run with the `stop` option.

When you run `init 2`, `init` runs the scripts in `/etc/rc.2`, one of which is `S75cron`. The script `S75cron` is executed with the `start` option as follows:

```
sh S75cron start.
```

Similarly, when you run `init 0`, `/sbin/rc0.d/K70cron` is executed with the `stop` option:

```
sh K70cron stop.
```

Running either of these scripts is the same as running `/sbin/init.d/cron` with the appropriate `start/stop` option.

Because these files are shell scripts, you can read them to see what they do. You can also change the files, although it is preferable to create your own versions because the delivered scripts may change in future releases.

Follow these rules when creating your own scripts:

- Place the file containing your script in the `/sbin/init.d` directory.
- Link the script (see `ln`(1)) to files in appropriate system state directories, using the naming convention described above.

Changing Default Boot Parameters

NOTE Before you change to a new default boot device, you must make sure that the disk is a valid bootable disk. See the procedure in this guide for making a device bootable before executing the `fltboot`(1M) command.

In the delivered AT&T 3B2 Computer, the default boot program name is NULL and the default bootable device is the first hard disk connected to the first disk controller. What this means is that when you are in firmware mode and attempt to boot the system, you will be prompted for a manual boot program name. After you enter the name of the program you want, the system will, by default, attempt to locate and load the program whose name you supplied from the default boot device.

You may change the default manual boot program to any bootable program name in the `/stand` directory using the `fltboot` command (see `fltboot`(1M)). Once you do this, the program name you select will be displayed in firmware mode as the default boot program. Similarly, you can change the default boot device that is booted on powerup or reboot.

The `fltboot` command is interactive, so no command line parameters are required. Two prompts are displayed, one for changing the default boot program and one for changing the default boot device. Pressing RETURN without entering a value at either prompt leaves the current value unchanged.

The following paragraphs explain how these defaults are displayed in firmware mode.

When you bring the system down to firmware mode (see "Changing to Firmware Mode," in this chapter) the following prompt is displayed:

```
        Enter name of program to execute [  ]:
```

Note that the brackets in the prompt are empty. This means that the default manual boot program name in Non-Volatile RAM (NVRAM) is null. If you change the default boot program using `fltboot`, the name you supply to `fltboot` will appear in the brackets shown above.

You have three options at this prompt:

- use `next part` to switch working partitions

- enter a program name and press RETURN
- press RETURN without entering a program name

The next part command changes the current working partition, if more than one is defined, on the default boot disk.

All these options cause the firmware to list the possible load devices connected to your machine and display the following prompt:

```
Enter Load Device Option Number [1 (HD70)]:
```

The brackets in this prompt show the default boot device for your machine, in this case a 70-Mbyte hard disk. You can enter a RETURN at this prompt, or choose another device and press RETURN. If you change the default boot device using `fltboot`, the device you supply to `fltboot` will appear in the brackets shown above.

What happens next depends on what appeared in the brackets in the previous prompt, and whether you entered the `next part` command or not.

If you had entered `next part`, a message appears telling you the current working partition, and the following prompt is displayed:

```
Enter name of boot file (or 'q' to quit):
```

If you did not enter `next part`, and either entered a program name or accepted the default contained in brackets, the system will boot from the current working partition using the name in brackets.

If you did not enter `next part`, and the brackets were empty, as shown above, the following prompt is displayed:

```
Enter name of boot file (or 'q' to quit):
```

If you press RETURN at this prompt (regardless of how you got to it), the contents of the current working partition is displayed, and the above prompt is repeated:

```
Enter name of boot file (or 'q' to quit):
```

At this point, you have the option of choosing one of the displayed programs, or entering `next part` to change the working partition to the next defined `stand` partition. If the currently displayed partition is the only one defined on the disk, then the system will tell you so, and you must then choose one of the displayed programs, or abort the boot procedure (enter q). Otherwise, entering `next part` again makes the first partition the working partition.

Changing to Firmware Mode

You must change to firmware mode to run any of the firmware programs that came with your machine. Firmware procedures include the use of the firmware-resident commands and the use of the bootable programs provided as part of the Essential Utilities. They might perform such functions as running diagnostics on system software, making a new floppy key disk, or changing the firmware password. See the hardware guide provided with your computer for a full description of the available firmware commands and instructions for running them.

To change to firmware mode, you must be logged in as `root` and follow these steps.

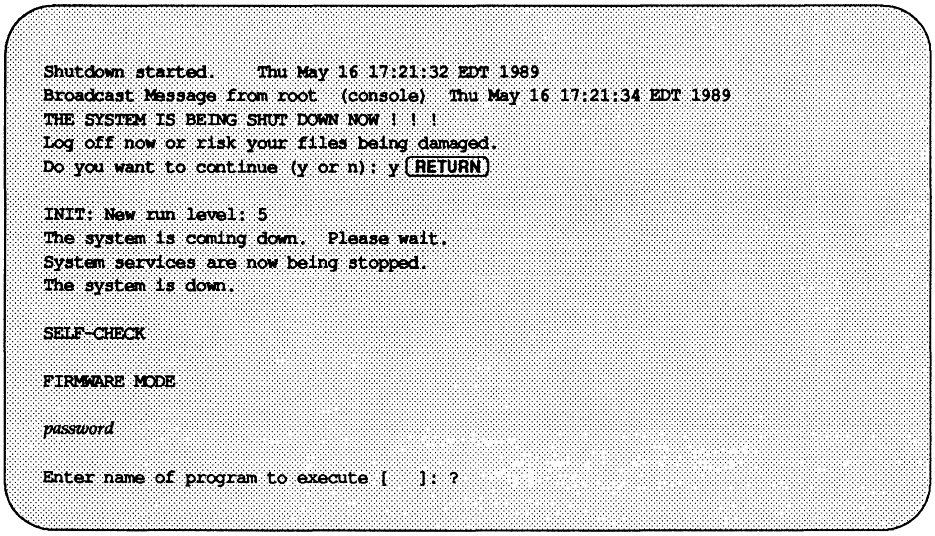
Step 1 Type:

```
shutdown -i5
```

A message appears on the console confirming that a shutdown has started. A message is also automatically broadcast that informs users of the shutdown and directs them to log off or risk their files being damaged.

Step 2 If other users are logged in, the system waits for a grace period of 60 seconds before asking you if you want to continue with the shutdown; answer `y` (for yes) at this prompt.

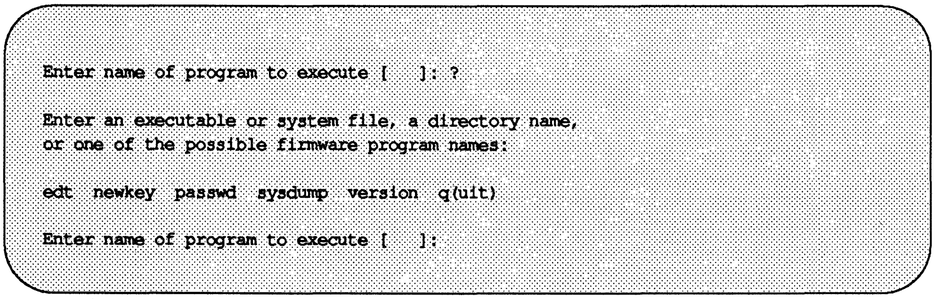
After you answer this question, the shutdown process starts and the system changes to firmware mode. The following screen shows an example of the system output displayed after you enter the `shutdown` command with other users logged in:



```
Shutdown started.   Thu May 16 17:21:32 EDT 1989
Broadcast Message from root  (console)  Thu May 16 17:21:34 EDT 1989
THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged.
Do you want to continue (y or n): y RETURN

INIT: New run level: 5
The system is coming down.  Please wait.
System services are now being stopped.
The system is down.

SELF-CHECK

FIRMWARE MODE

password

Enter name of program to execute [   ]: ?
```

The default firmware password is mcp.

Step 3 You are now ready to run the firmware programs available on your computer. To display the firmware program menu, enter a question mark (?) in response to the message `Enter name of program to execute [ ]`, that is displayed when you enter firmware mode. The following screen shows how to display the firmware program menu.



```
Enter name of program to execute [   ]: ?

Enter an executable or system file, a directory name,
or one of the possible firmware program names:

edt  newkey  passwd  sysdump  version  q(uit)

Enter name of program to execute [   ]:
```

See the hardware guide for your computer for specific instructions on running firmware programs.

See the "Returning from Firmware" section in this chapter for information on returning to an operating system state from firmware.

Creating a Floppy Key

The floppy key is a floppy disk that contains a copy of your computer serial number and certain information contained in Non-Volatile Random Access Memory (NVRAM). The floppy key can be used to access the operating system in the event that the system administrator forgets the firmware password or in the unlikely event that information in the NVRAM is corrupted.

The floppy key resets the following system parameters to their original values: the system name, node name, speed of the console terminal, and the firmware password. The floppy key is typically used to recover from a forgotten firmware password.

A procedure for making a floppy key can be found in the ''System Setup'' chapter; it is one of the first things you should do when you setup your system.

Powering Down Your Machine

The powerdown procedure differs depending on whether the system is in multi-user or single-user state.

From Multi-User State

If your system is in multi-user state, turn off your computer by running the `shutdown -i0` command. This command flushes the system buffers, closes any open files, stops all user processes and daemons currently running, unmounts file systems, and then removes power from the computer.

CAUTION Do not pull the plug until the powerdown procedure is completely finished and you can no longer hear the disks or fans.

1. Before powering down your system, check to see who is logged on at the time by executing `who -H`, as shown in the example below.

```
$ who -H
NAME      LINE      TIME
rht       term/12   Aug 31 10:28
mark      term/14   Aug 31 09:43
root      console   Aug 31 08:32
$
```

   If there are any users logged in on the system, go to step 2; if not, go to step 3.

2. To notify any users that the system is about to be shut down, type `wall` followed by a message announcing the shutdown. To end your message, type CTRL-d as shown in the following example.

```
$ wall
The system will be coming down in 5 minutes.
Please log off. CTRL-d
Broadcast Message from root (console) on unix Wed Feb 26 07:30:27...
The system will be coming down in 5 minutes.
Please log off.
$
```

3. If there are no users logged on, or after you have notified all users who are logged on, power down the system by executing:

   `shutdown -i0`

   The next screen shows an example of the system output displayed at this time.

```
Shutdown started.    Thu May 16 17:10:57 EDT 1989

Broadcast Message from root (console)  Thu May 16 17:10:59
THE SYSTEM IS BEING SHUT DOWN NOW ! ! !
Log off now or risk your files being damaged.
Do you want to continue (y or n): y RETURN

INIT: New run level: 0
The system is coming down.  Please wait.
System services are now being stopped.
The system is down.
```

By now the power has been removed, cabinet lights turned off, the fan(s) stopped, and the hard disk(s) stopped spinning.

To protect your system from being turned off by an unauthorized user, assign a password to the `powerdown` command (see "Assigning Special Administrative Passwords" in the "Security" chapter).

From Single-User State

CAUTION Do not pull the plug until the powerdown procedure is completely finished and you can no longer hear the disks or fans.

If your system is in single-user state, turn off your computer by running the `shutdown` command as follows:

```
shutdown -y -i0 -g0
```

where the arguments are defined as:

`-y`	answer yes to all questions asked by `shutdown`
`-i0`	change the system to state 0 (off)
`-g0`	allow a grace period of 0 seconds (for you to log off)

Console messages will appear as shown below.

```
Shutdown started.    Mon Jul  3 12:17:57 EDT 1989

INIT: New run level: 0
The system is coming down.  Please wait.
System services are now being stopped.
The system is down.
```

Shortly after the last message in the screen above appears, all system services stop and power is removed from the machine.

Rebooting Your System

This procedure halts the system and either reboots from the bootable operating system currently on the hard disk, `/stand/unix`, or configures a new bootable operating system (if necessary) and reboots from the new `/stand/unix`.

This method forces a configuration of a new bootable operating system, if required because of software modifications to the system. (See "Configuring the UNIX Operating System," in the "Performance Management" chapter, for more about configuring a new bootable operating system.) If a new bootable operating system is created, it is written to `/stand/unix`, and the system is rebooted using the new bootable operating system.

To halt and reboot the system from the hard disk, you must be logged in as root. Enter:

```
shutdown -i6
```

A message appears on the console confirming that a shutdown has started. A message is also automatically broadcast that informs users of the shutdown and directs them to log off or risk their files being damaged. The system then waits for a grace period of 60 seconds before asking you if you want to continue with the shutdown; answer `y` (for yes) at this prompt.

The messages shown below assume that no reconfiguration of the operating system is necessary. Should a reconfiguration take place, the messages will look different after the change to the new run level. See "Configuring the UNIX Operating System," in the "Performance Management" chapter.

```
Shutdown started.  Mon Nov 21  10:32:02 EST 1988
Broadcast Message from root (console) Mon Nov 21 10:32:02
THE SYSTEM IS BEING SHUTDOWN NOW ! ! !
Log off now or risk your files being damaged.
Do you want to continue (y or n): y RETURN

INIT: New run level: 6
The system is coming down.  Please wait.
System services are now being stopped.
The system is down.
The system is being restarted.
SELF-CHECK
```

At this point, the messages shown are exactly those that appear when the system is powered up from a halted state, with the exception that diagnostics are not run. The system is placed in the state defined by the `initdefault` entry in `/sbin/inittab`. If this entry specifies multi-user state, you receive the prompt:

```
        Console Login:
```

Note that it may take 10 minutes or longer, depending on your machine model and equipment, to get to the `Console Login:` prompt after the `SELF-CHECK` message appears.

After you receive the `Console Login:` prompt, you may log in to your rebooted system.

Displaying Summary Configuration Information

To print system configuration information, type:

```
prtconf
```

The system will display information about memory and peripheral configuration, such as the information shown in the following example:

```
AT&T 3B2 SYSTEM CONFIGURATION:

Memory size: 4 Megabytes
System Peripherals:

        Device Name     Subdevices           Extended Subdevices

        SBD
                        Floppy Disk
                        72 Megabyte Disk
                        72 Megabyte Disk
        PORTS
        CTC
        NAU
        PORTS
        MAU
```

The device and subdevice names displayed with this command are the AT&T administered names found in the Equipped Device Table (EDT) (see `edittbl`(1M)).

Displaying System Name and Operating System Release Number

To display your system name and the number of your UNIX operating system release, use the uname command with the `-s` and `-r` options. For example:

```
$ uname -sr
UNIX System V 4.0
```

In this example, the name of the system is `unix` and the release number of the UNIX system being run is `4.0`. See uname(1) for a list of other information you can display and change with the uname command.

Displaying Who Is Logged on to Your Machine

Before taking any action that would affect a system user, check to see who is logged on to the system. The who command displays a list of users logged on to your machine, along with the ID, terminal number, and login time of each user. Using the -H option supplies you with headers to the information. For example:

```
$ who -H
NAME     LINE       TIME
root     console    Aug 31 08:32
opns     term/13    Aug 30 21:28
mrdh     term/12    Aug 31 09:43
$
```

The who command has a number of options that allow you access to more information than the previous example shows. The following list describes some of these options. For a complete listing and a more detailed explanation of each, see who(1) in the *User's Reference Manual*.

-u	On each line, show the number of hours and minutes since activity last occurred. A dot (.) indicates that the terminal has been active in the last minute and is therefore "current." The information is included as an additional field on the default display for the who command.
-T	Show whether someone else can write to that terminal. A plus sign (+) appears if the terminal is writable by anyone; a minus sign (–) appears if it is not. root can write to all terminal lines. If a bad line is encountered, a ? is printed.
-l	Show lines on which the system is waiting for someone to log in.
-q	Show a "quick" listing, containing only the user login for those who are logged on to the system and the total number of users logged on.
-b	Show date and time of the last reboot.

`-r`	Show the current system state of the init process.
`-a`	Run who with all options turned on.

Returning from Firmware

After firmware programs have been run, you can bring the system back from the firmware mode by executing `unix` from the hard disk.

Step 1 When you receive the firmware prompt, enter `unix`:

```
Enter name of program to execute [ ]: unix
```

Step 2 At the next prompt, press the RETURN key to boot the operating system from the default boot disk (shown in brackets).

```
Possible load devices are:
Option Number    Slot    Name
------------------------------
      0            0    FD5
      1            0    HD72
      2            0    HD72
      3            0    CTC
Enter Load Device Option Number [1 HD 30]: RETURN
```

Step 3 After the sanity of the root file system is checked, file system checks are performed as necessary, the system configuration is printed out, and the system is placed in the operating state defined by the `init-default` entry in `/sbin/inittab`. If this state is multi-user state, you will eventually observe the prompt:

```
Console Login:
```

Note that it may take 10 minutes or longer, depending on your machine model and equipment, to get to the `Console Login:` prompt.

After you receive the `Console Login:` prompt, you may log in to your system.

Making New Bootable Disks

Each time you turn on your computer, reboot, or manually request a boot from firmware mode, the programs and data files used to boot the computer are read from the hard disk or the floppy disk and loaded into memory for execution. For normal operations, you need to know little more. You know where the programs reside (in `/stand`) and you know how to reconfigure some of these programs (like the bootable operating system) should the configuration of your computer change.

This section shows you how to make additional bootable hard disks, should the need arise. For example, you may want to make another bootable hard disk for development purposes.

After making a new bootable disk, you can bring your machine down to the firmware level (`shutdown -i5`) and request a boot from the new bootable disk.

Making a New Bootable Hard Disk

CAUTION This procedure should be used by experienced system administrators only. Users unfamiliar with disk partitioning, multi-disk operations, and the operation of the system in both firmware mode and during the boot process should not attempt this procedure.

During the process of installing the system software on your computer, the disk attached to the first hard disk controller is made a bootable disk and becomes the default hard disk for the boot process. This disk has a pathname of the form `/dev/rdsk/c1d0s6`, for the character special file (raw device), and `/dev/dsk/c1d0s6`, for the block special file (block device).

Whenever you power up or reboot your computer, the disk described above is the disk from which the boot programs, associated data files, and the bootable operating system are taken. The computer loads the boot program found in the `boot` partition of the default hard disk. The boot program will then load the bootable operating system found in the boot file system in the first `stand` partition on the default hard disk. (In most delivered systems, the default hard disk is the first integral hard disk connected to the first disk controller.)

When the system is in firmware mode, you can request a reboot from any bootable hard disk connected to your system, regardless of which device contains the root file system.

If your computer has more than one hard disk, you can make any disk a bootable disk.

A bootable disk must have at least two partitions defined on it:

- It must have a `boot` partition. This partition holds the `boot` program that loads and executes the bootable operating system.
- It must have a `stand` partition that has a file system of type `bfs` defined on it. This boot file system contains all the bootable programs for your computer, as well as all data files needed by these programs. The boot file system is like other file systems in that it can be mounted and unmounted under any directory. By default it is mounted as `/stand`.

You can make any disk connected to your computer bootable, and either boot that disk from firmware, or make it the default boot disk. However, you should be aware that maintaining two or more bootable devices, depending on your intentions, may involve constantly mirroring changes made in one boot file system to other boot file systems. If you make changes to your machine's configuration, such as adding a new hardware or software module, changing tunables, etc., you may want to copy the bootable operating system to all boot file systems, so that each contains a valid bootable operating system for your machine's current configuration.

For example, suppose you have a two-disk system, with boot file systems and boot partitions on both disks (Disk A and Disk B). Assume that the boot file system on Disk A is mounted as `/stand` and the boot file system on Disk B is mounted as `/stand2`. Disk A is the disk booted on powerup and on a `shutdown -i6`. Let's say that you install a new software driver, which entails changing the `/stand/system` file on Disk A, and reboot the machine.

When the machine reboots, the firmware will detect a difference in the `/stand/system` file on Disk A, and will cause the configuration of a new bootable operating system (`/stand/unix`) on Disk A. After the configuration completes, the machine boots from Disk A.

Now you have two different bootable operating systems on the two disks; the one on Disk A knows about the new software driver you installed, while the one on Disk B does not.

A similar situation arises whenever you make other kinds of changes to the configuration of your system, such as the addition or removal of expansion boards, device drivers, and the like. It is a good idea to copy the bootable operating system to other boot file systems whenever you make changes to the configuration of your system.

In cases like this, request a boot from each defined boot file system on each disk. To do this, request a boot of the `/stand/system` file from firmware and then choose the appropriate disk.

Note that the boot program examines the Volume Table of Contents (VTOC) on each disk to find the boot file system. The boot file system used depends on which disk you request at the firmware prompt (on powerup and reboot, the default boot disk specified in Non-Volatile RAM is assumed). If you define multiple boot file systems on a disk (i.e., multiple `stand` partitions), you can use the `next part` firmware command to boot from a specific boot file system (see "Operations on the stand Partition," in this chapter, for a description of the `next part` command).

Also note that the file systems mounted when the system comes up in multi-user state are specified in `/etc/vfstab`.

The following procedure shows how to make a hard disk bootable.

Step 1 If the hard disk you want to make bootable has data on it, perform a full backup of the hard disk. You will want to reload this data after you make the disk bootable. See the "Backup Service" chapter for backup instructions.

Step 2 Use the `prtvtoc`(1M) command to list the partitions currently on the disk. In order to make the disk bootable, `boot` and `stand` partitions must exist on the disk and must have the following minimum sizes: `boot`, 100 512-byte sectors; `stand`, 5500 512-byte sectors. The command looks like the following:

```
prtvtoc /dev/rdsk/c?d?s0
```

where the question marks in the device name are replaced by the appropriate controller and drive number, respectively. See `prtvtoc`(1M) to identify the partitions on the disk from the output.

Step 3 If Step 2 reveals that the boot and stand partitions do not exist, or do not meet the minimum sizes specified in Step 2, you must repartition the hard disk.

At a minimum, the disk must be formatted to contain boot and stand partitions, as well as a partition (partition 6) that contains the whole disk. Other partitions can be defined on remaining space on the disk as needed, including creating additional stand partitions.

You can partition a disk using either a restore procedure in the hardware guide for your computer, the fmthard(1M) command, or the sysadm devices menu.

See the "Storage Device Management" chapter for a chart of default hard disk partitions for various disk drives. If your disk is not in the chart, select the one that most closely matches yours. Then use the values you find in the chart for the boot and stand partitions to repartition your disk.

Step 4 Make a boot file system on the stand partition using mkfs(1M). The command looks like the following:

```
mkfs -F bfs /dev/rdsk/c?d?s? 5500 [#inodes]
```

where the question marks in the device name are replaced by the appropriate controller, drive, and partition numbers, respectively. The *#inodes* parameter is optional; it specifies the maximum number of files permitted in the boot file system. If omitted, the system will choose a number of files based on the number of blocks specified. Under most circumstances, this parameter may be omitted. See mkfs(1M) and the "File System Administration" chapter for complete information on making a boot file system.

Step 5 Mount the new /stand file system as /mnt:

```
mount -F bfs /dev/dsk/c?d?s? /mnt
```

Replace the question marks with appropriate controller, drive, and partition numbers, as above.

Step 6 Copy the contents of /stand on the old bootable disk to /mnt. Use any copy method you like, but the following is recommended:

```
cd /stand
find . -type f -print | cpio -pumv /mnt
```

Note that you may have to raise the allowable file size limit using the `ulimit` shell command (see `sh`(1)) if `/stand/unix` is larger than the current maximum file size limit.

Step 7 Use `umount`(1M) to unmount the boot file system from `/mnt`, as follows:

```
umount /mnt
```

Step 8 Copy the boot programs from the old bootable disk to the new boot partition on the new bootable disk using the `newboot`(1M) command, as follows:

```
newboot /usr/lib/boot /usr/lib/mboot \
        /dev/rdsk/c?d?s7
```

Replace the question marks with appropriate controller and drive numbers. The partition number specified must be 7.

Step 9 Make any other file systems desired on remaining disk partitions. If you performed a backup in Step 1, restore data from the backup to a file system on the new bootable disk. See the "Restore Service" chapter for descriptions of restore methods.

Step 10 Edit `/etc/vfstab` and `/etc/boot_tab` to match the new file system layout on your machine. These files determine the file systems that are mounted at boot time and where they are mounted. Use `prtvtoc`(1M) to list the partitions/file systems on all the disks, and compare this output to the above files. Then make changes so that the file systems you expect to be mounted at boot time are specified in these files.

You have now defined another bootable disk for your system. Right now, the only way to boot from this new disk is to explicitly request a boot from the disk from firmware. If you want this disk to become the default disk used at powerup and reboot, you must follow the instructions in the section "Changing Default Boot Parameters" in this chapter.

Quick Reference to Machine Management

- Changing to firmware mode:

  ```
  shutdown -i5
  ```

  shuts the system down. The -i5 option places you in firmware mode. When asked at what intervals warning messages should be given, answer n (for no) to shutdown the system with only a short delay. When you see the message FIRMWARE MODE you may begin running firmware programs.

- Powering down your machine from multi-user state:

  ```
  shutdown -i0
  ```

  shuts down system power. Before executing shutdown(1M), use who(1) to check who is logged on and wall(1M) to notify those users of your intentions to power down the system.

- Powering down your machine from single-user state:

  ```
  shutdown -y -i0 -g0
  ```

  where the -y option assumes "yes" to all questions, -i0 shuts down the system to state 0 (meaning off), and -g0 defines the grace period as 0 seconds. You can use -y and -g0 in this case since you are the only user on the machine in single-user state.

- Rebooting your system:

  ```
  shutdown -i6
  ```

  where -i6 shuts down the system to state 6, meaning stop the system and reboot.

- Displaying summary configuration information:

  ```
  prtconf
  ```

  produces a display which includes memory and peripheral configuration information.

- Displaying system name and operating system release number:

  ```
  uname -sr
  ```

  displays your system name (e.g., unix) and UNIX operating system release number.

- Displaying who is logged on to your machine:

  ```
  who -H
  ```

  produces a display of users logged on to your machine and shows the ID, terminal number, and sign-on time of each user. The `-H` option adds field headers to the display.

7. NETWORK SERVICES

7 Network Services

Introduction

This chapter describes the administrative command-level interface for Network Selection and the Basic Networking Utilities (BNU). Distributed File Systems (RFS and NFS) are documented in the *Programmer's Guide: System Services and Application Packaging Tools*.

Network Selection and the Service Access Facility (SAF) grew out of the need for UNIX Systems to communicate both with other UNIX systems and with non-UNIX systems. Previous to UNIX System V Release 4, it was difficult for network applications to find out what transport providers were available on a given machine and to find the addresses of services. The Network Selection mechanism and the Name-to-Address Mapping facility described in this section provide both a consistent way to determine what transport providers are installed on a machine and a standardized mechanism for finding service addresses.

The new Network Selection feature described in this chapter generalizes the procedure by which an application chooses the network it connects to. This procedure is implemented as a set of library routines for inclusion in application programs. The system administrator is responsible for maintaining the network configuration database file (`/etc/netconfig`) used by these routines. He/she is also responsible for creating and maintaining the "host" and "service" files required for each of the Name-to-Address Mapping libraries.

The "Network Selection" section of this chapter describes the `netconfig` file and the `NETPATH` environment variable, which may be used by both users and the system administrator to customize the default list of networks an application tries to connect to. The "Name-to-Address Mapping" section describes the Name-to-Address Mapping files that must be in place before applications can use the Name-to-Address Mapping libraries. The library routines available to application programmers are described in the *Programmer's Guide: Networking Interfaces*, on the `getnetconfig`(3N) manual page and on the `netdir`(3N) manual page.

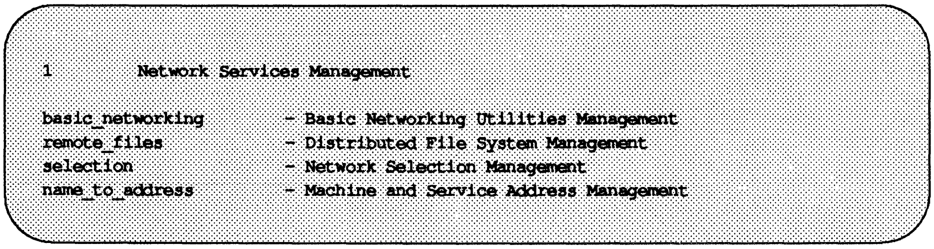
The remaining sections of this chapter describe the Basic Networking Utilities (BNU).

You can do any of the functions associated with Networking Services administration by selecting the appropriate "task" from a series of menus provided for administration. To access the "system administration" menu for using Networking Services, type:

```
        sysadm network_services
```

The following menu will appear on your screen:

Figure 7-1: Network Services Management Menu



```
1         Network Services Management

basic_networking         - Basic Networking Utilities Management
remote_files             - Distributed File System Management
selection                - Network Selection Management
name_to_address          - Machine and Service Address Management
```

NOTE The `Distributed File System Management` option is described in the *Programmer's Guide: System Services and Application Packaging Tools*.

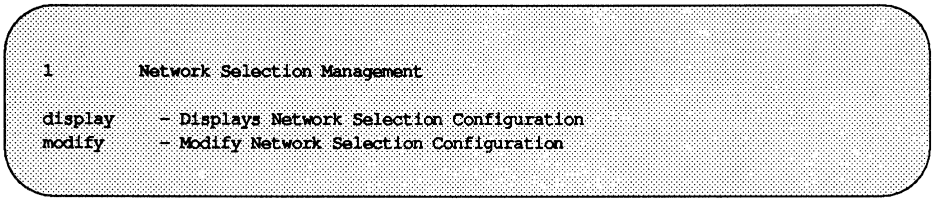
When you have selected the option you want, the submenus and instructions displayed on the screen are self-explanatory and lead you through the appropriate procedures.

Network Selection

In order for network applications to be portable to different environments, the application process must have a standard interface into the various networks available in any current environment. Network Selection provides a simple and consistent interface that allows user applications to select networks (at the transport level), enabling applications to be protocol- and media-independent. System V Networking Services applications that allow the user to influence the choice of networks use the standard interface outlined here.

Tasks associated with Network Selection administration may be performed using either the menu system or shell commands entered on the command line. The screen below is the top-level menu for Network Selection. It can be brought up on the screen by typing `sysadm selection`.

Figure 7-2: Network Selection Management Menu



```
1          Network Selection Management

display    - Displays Network Selection Configuration
modify     - Modify Network Selection Configuration
```

When you select an option, self-explanatory submenus and instructions lead you through the appropriate procedures.

You can also bypass the menu system by issuing commands directly to the shell. Where these commands involve editing sensitive system files, be sure to keep a backup copy of the file you are editing. When you have finished editing the file, use `diff`(1) on the edited file and the backup copy to verify that only the changes you want have been made.

Table 7-1: Command Alternatives to the Network Selection Management Menu

Task Description	Menu Item	Shell Command
Add networks to the network configuration database file `/etc/netconfig`.	add	`vi /etc/netconfig`

Display the contents of the `netconfig` file; display the entry for network *netid*.	display	`cat /etc/netconfig`
		`grep` *netid* `/etc/netconfig`

Change a `netconfig` entry	modify	`vi /etc/netconfig`
Delete a `netconfig` entry.	remove	`vi /etc/netconfig`

Network Selection Overview

The UNIX System V Release 4.0 Network Selection component is built around

- a network configuration database (the `/etc/netconfig` file) that contains entries for each network available to the system, and
- an optional `NETPATH` environment variable, set by a user or the system administrator and containing an ordered list of network identifiers. These network identifiers match the `netconfig` *network ID* field and are used as links to the records in the `netconfig` file.

The Network Selection application programming interface consists of a set of network configuration database access routines. One group of these library routines accesses only the `netconfig` entries identified by the `NETPATH` environment variable; another group of routines accesses `netconfig` directly. The routines are described in the *Programmer's Guide: Networking Interfaces*. The first group is also described in detail in the manual page `getnetpath`(3N). The second group is described in `getnetconfig`(3N).

Applications should use the routines that access `NETPATH`. They allow users to influence the selection of transports used by the application. If an application does not want the user to influence its decision, then the routines that access the `netconfig` database directly should be used.

The `netconfig` file, on which the Network Selection library routines depend, is maintained by the system administrator. The `NETPATH` environment variable is typically set or modified by application programmers and users, depending on the needs of their applications, but it may also be set by the system administrator in response to the needs of administrative applications.

The `netconfig` File

The system administrator is responsible for maintaining the network configuration database file `/etc/netconfig`. Entries in the `netconfig` file contain the following fields, in the order shown:

Table 7-2: Fields in `netconfig` Entries

network ID	*semantics*	*flag*	*protocol family*	*protocol name*	*network device*	*directory lookup libraries*

The fields correspond to elements of the `struct netconfig` structure. Pointers returned by Network Selection library routines are pointers to `netconfig` entries in `struct netconfig` format. The `netconfig` file is described in the manual page `netconfig`(4). The `netconfig` manual page also describes the elements of the `struct netconfig` structure. All symbolic names, structure definitions, and constant values for the Network Selection feature are defined in the header file `/usr/include/netconfig.h`.

`netconfig` fields are defined as follows:

network ID

A string used to identify a network. *network ID* consists of non-NULL characters, and has a length of at least 1. No maximum length is specified. This namespace is locally-significant and the local system administrator is the naming authority responsible for ensuring that all *network IDs* on a system are unique.

semantics

A string that identifies the "semantics" of the network, that is, the set of services it supports, by identifying the service interface it provides. This is closely related to, but not identical with, the API (Application Programming Interface) with which applications are "supposed" to access the network. Typically, an application will specify its API by pushing an appropriate STREAMS module (such as `timod`) and using an appropriate user-level library (such as the TLI library). The *semantics* field is mandatory. The following semantics are recognized.

`tpi_clts`	Transport Provider Interface, connectionless
`tpi_cots`	Transport Provider Interface, connection-oriented
`tpi_cots_ord`	Transport Provider Interface, connection-oriented and supports orderly release

flag

The *flag* field records certain two-valued ("true" and "false") attributes of networks. *flag* is a string composed of a combination of characters, each of which specifies the value of the corresponding attribute. If the character is present, the attribute is "true." If the character is absent, the attribute is "false." A hyphen (–) specifies that none of the attributes is present. Only one character is currently recognized:

`v`	Visible ("default") network. Used when the environment variable `NETPATH` is *unset.*

protocol family

The *protocol family* and *protocol name* fields are provided for protocol-specific applications. The *protocol family* field contains a string that identifies a protocol family. The *protocol family* identifier follows the rules for *network IDs*, which are:

- The string consists of non-NULL characters.
- It has a length of at least 1.
- There is no maximum length specified.

A hyphen (–) in the *protocol family* field indicates that none of the available protocol family identifiers applies, that is, the network is experimental. An application that wants to have family characteristics can match on the *protocol family* field when selecting a network (for example, an application can search for an "osi" family). In this case, the application is not protocol independent, since it has searched only for OSI entries. The following are examples of protocol family identifiers:

`loopback`	Loopback (local to host)
`inet`	Internetwork: UDP, TCP, etc.
`implink`	ARPANET imp addresses
`pup`	PUP protocols: for example, BSP
`chaos`	MIT CHAOS protocols
`ns`	XEROX NS protocols
`nbs`	NBS protocols
`ecma`	European Computer Manufacturers Association
`datakit`	DATAKIT protocols
`ccitt`	CCITT protocols, X.25, etc.
`sna`	IBM SNA
`decnet`	DECNET
`dli`	Direct data link interface
`lat`	LAT
`hylink`	NSC Hyperchannel

`appletalk`	Apple Talk
`nit`	Network Interface Tap
`ieee802`	IEEE 802.2; also ISO 8802
`osi`	Umbrella for all families used by OSI (for example, `protosw` lookup)
`x25`	CCITT X.25 in particular
`osinet`	AFI = 47, IDI = 4
`gosip`	U.S. Government OSI

protocol name The *protocol name* field contains a string that identifies a protocol. This field is currently only used for the `inet` family. For any other family, the protocol name field contains a hyphen (–). The *protocol name* identifier follows the same rules as *network IDs*:

- The string consists of non-NULL characters.
- It has a length of at least `1`.
- There is no maximum length specified.

The *protocol name* field may contain:

`icmp`	Internet Control Message Protocol
`tcp`	Transmission Control Protocol
`udp`	User Datagram Protocol

network device The *network device* is the full pathname of the device used to connect to the transport provider. Typically, this device will be in the `/dev` directory. The *network device* must be specified.

directory lookup libraries
The *directory lookup libraries* support a "directory service" (that is, a Name-to-Address Mapping service) for the network. This service is implemented by the UNIX System V Name-to-Address Mapping feature. If a network is not provided with such a library, the Name-to-Address Mapping feature will not work. A hyphen (–) in this field shows that lookup libraries, and therefore Name-to-Address Mapping, are unavailable.

The *directory lookup library* field consists of a comma-separated list of full pathnames to dynamically linked libraries. Literal commas may be embedded as "\,"; backslashes as "\\". Lines in `/etc/netconfig` that begin with a pound sign (#) in column 1 are comments.

The system administrator determines the *order* of the entries in the `netconfig` database. Since the Network Selection library routines that access `netconfig` directly return entries in order, beginning at the top of the `/etc/netconfig` file, the order in which networks are entered in the file by the system administrator becomes the default search path for applications choosing networks to connect to.

Figure 7-3: Sample `netconfig` File

```
starlan    tpi_cots      v osinet    -   /dev/starlan   /usr/lib/straddr.so
starlandg  tpi_clts      v osinet    -   /dev/starlandg /usr/lib/straddr.so
npack      tpi_cots      v localnet  -   /dev/npack     /usr/lib/npack.so
tcp        tpi_cots_ord  v inet      tcp /dev/tcp       /usr/lib/tcpip.so
udp        tpi_clts      v inet      udp /dev/udp       /usr/lib/tcpip.so
ticlts     tpi_clts      v loopback  -   /dev/ticlts    /usr/lib/straddr.so
ticots     tpi_cots      v loopback  -   /dev/ticots    /usr/lib/straddr.so
ticotsord  tpi_cots_ord  v loopback  -   /dev/ticotsord
```

The `NETPATH` Environment Variable

Network Selection is unobtrusive. In most cases the user isn't interested in which network handles a network operation, and the default network search path established by the system administrator (the `netconfig` file) is used to locate a network available for connection. However, if a user or the system administrator wants to influence the choices made by applications, the search path can be modified using a new standard shell variable, `NETPATH`. `NETPATH` is similar to the `PATH` variable.

NETPATH consists of a colon-separated list of network IDs. Each network ID corresponds to the *network ID* field of a record in the netconfig database. A literal colon can be embedded as "\:", and a literal backslash as "\\". An empty component in NETPATH, signified by either a beginning colon, an ending colon, or two successive colons, is not a valid entry, since the empty string is not a valid network ID. NETPATH is described in the environ(5) manual page in the *User's Reference Manual.*

The NETPATH environment variable is not set in /etc/profile. It can, however, be set in a user's $HOME/.profile.

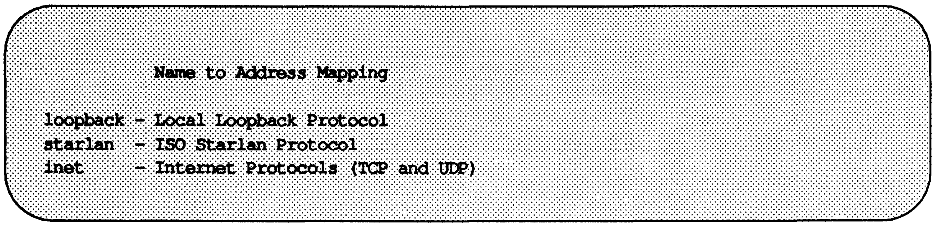
Users and system administrators alike should be aware that the set of "default" networks is different for the routines that access netconfig directly and the routines that access netconfig via the NETPATH environment variable. For the routines that access netconfig directly, the set of default networks is the entire netconfig file; the set of "default" networks for the routines that access netconfig via NETPATH is the "visible" networks in the netconfig file. A network is "visible" if the system administrator has included a v flag in the flag field. If NETPATH is unset, these "visible" networks are the default search path for this second group of access routines.

Name-to-Address Mapping

The Name-to-Address Mapping feature allows an application to obtain the address of a service on a specified machine in a transport-independent manner.

Tasks associated with Name-to-Address Mapping administration may be performed using the menu system or shell commands entered on the command line. The screen below is the top-level menu for Name-to-Address Mapping. It can be brought up on the screen by typing `sysadm name_to_address`.

Figure 7-4: Machine and Service Address Management Menu



```
            Name to Address Mapping

loopback - Local Loopback Protocol
starlan  - ISO Starlan Protocol
inet     - Internet Protocols (TCP and UDP)
```

When you have selected the option you want, self-explanatory submenus and instructions will be displayed on the screen to lead you through the appropriate procedures.

If you want to bypass the menu system, you may issue commands directly to the shell, as shown in Table 7-3.

Table 7-3: Shell Commands for Name-to-Address Mapping

Task Description	Menu Item	Shell Command

Local Loopback Protocol
ISO Starlan Protocol
	`loopback`
	`starlan`
		`vi /etc/net/`*transport*`/hosts`
		`vi /etc/net/`*transport*`/services`

Internet Protocols (TCP and UDP)	`inet`	`vi /etc/hosts`
		`vi /etc/services`

Name-to-Address Mapping consists of routines for use by application programs. These routines (which are described on the `netdir`(3N) manual page) are used to obtain addresses of services on given hosts. All routines are combined into a library, one for each transport provider. The library to use for a specific transport provider is named in the `/etc/netconfig` file in the entry for each network. The `netdir_getbyname()` routine dynamically links the library named in the *directory lookup libraries* field of the `/etc/netconfig` file.

The routines are:

```
netdir_getbyname
netdir_getbyaddr
netdir_free
netdir_mergeaddr
taddr2uaddr
uaddr2taddr
netdir_options
```

Each function takes a pointer to a `netconfig` structure and returns a list of addresses of the service and host names over a given transport provider.

Both libraries provide all the above routines. The libraries are:

`tcpip.so`	contains the Name-to-Address Mapping routines for the TCP/IP protocol suite
`straddr.so`	contains the Name-to-Address Mapping routines for any protocol that accepts strings as addresses ISO, STARLAN, and the loopback drivers are examples.

Setting Up the Name-to-Address Mapping Libraries

Files for each of the libraries must be created and maintained by the system administrator.

`tcpip.so.`

The routines in this dynamic library create addresses from the `/etc/hosts` and `/etc/services` files available with the TCP/IP package. The `/etc/hosts` file contains two fields, the machine's IP address and the machine name. For example:

```
192.11.108.01       bilbo
192.11.108.16       elvis
```

The `/etc/services` file contains two fields, a service name and a port number with one of two protocol specifications, either `tcp` or `udp`. For example:

```
rpcbind      111/udp
rpcbind      111/tcp
login        513/tcp
listen       1025/tcp
```

For an application to use this library to request the address of a service on a particular host, the host name must appear in the `/etc/hosts` file and the service name must appear in the `/etc/services` file. If one or the other does not appear, an error will be returned by the name to address mapping routines.

`straddr.so`.

The routines in this dynamic library create addresses from files that have the same format as the `tcpip.so` file described above. The `straddr.so` files are `/etc/net/`*transport*`/hosts` and `/etc/net/`*transport*`/services`. *transport* is the local name of the transport provider that accepts string addresses (specified in the *network ID* field of the `/etc/netconfig` file). For example, the host file for `starlan` would be `/etc/net/starlan/hosts`, and the service file for `starlan` would be `/etc/net/starlan/services`. For `starlandg`, the files would be `/etc/net/starlandg/hosts` and `/etc/net/starlandg/services`.

Even though most string addresses do not distinguish between "host" and "service," separating the string into a host part and a service part provides consistency with other transport providers. The `/etc/net/`*transport*`/hosts` file will therefore contain a string that is considered to be the machine address, followed by the machine name. For example:

```
bilboaddr     bilbo
elvisaddr     elvis
frodoaddr     frodo
```

The /etc/net/*transport*/services file contains a service name followed by a string identifying the service port. For example:

```
rpcbind       rpc
listen        serve
```

The routines create the full string address by combining the "host address" and the "service port," separating the two with a dot (.). For example, the address of the "listen" service on bilbo would be bilboaddr.serve and the address of the "rpcbind" service on bilbo would be bilboaddr.rpc.

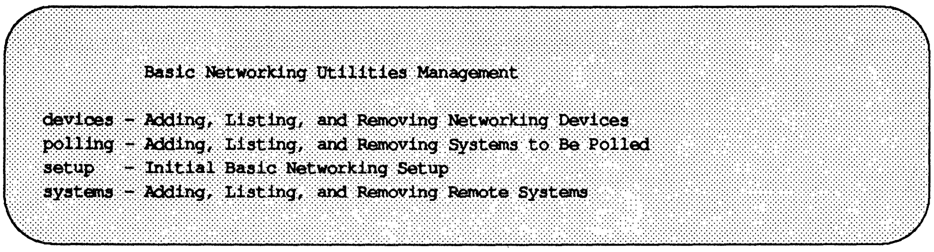
When an application requests the address of a service on a particular host on a transport provider that uses this library, the host name must appear in /etc/net/*transport*/hosts and the service name must appear in /etc/net/*transport*/services. If one or the other does not appear, the Name-to-Address Mapping routines return an error.

Basic Networking Utilities

The Basic Networking Utilities Package (BNU) lets computers using the UNIX operating system communicate with each other and with remote terminals. These utilities range from those used to copy files between computers (uucp and uuto) to those used for remote login and command execution (cu, ct, and uux).

Tasks associated with Basic Networking Utilities administration may be performed using either the menu system or shell commands entered on the command line. The screen below is the top-level menu for BNU selection. It can be brought up on the screen by typing `sysadm basic_networking`.

Figure 7-5: The Basic Networking Utilities Management Menu



```
              Basic Networking Utilities Management

devices - Adding, Listing, and Removing Networking Devices
polling - Adding, Listing, and Removing Systems to Be Polled
setup   - Initial Basic Networking Setup
systems - Adding, Listing, and Removing Remote Systems
```

When you have selected the option you want, the self-explanatory submenus and instructions will be displayed on the screen to lead you through the appropriate procedures.

If you want to bypass the menu system, you may issue equivalent commands directly to the shell as shown in Table 7-4:

Table 7-4: Shell Commands for Basic Networking Utilities

Task Description	Menu Item	Shell Command
Add, list, and remove networking devices	`devices`	`vi /etc/uucp/devices`
Add, list, and remove systems to be polled	`polling`	`vi /etc/uucp/poll`
Set up basic networking	`setup`	*use menu item*
Add, list, and remove remote systems	`systems`	`vi /etc/uucp/systems`

In this chapter, you will first get a high level introduction to the process of Basic Networking, as well as an introduction to the components involved in the process (network hardware, commands, daemons, and support files). Keep this high level view in mind as you begin to work with BNU; it will help you keep track of where you are in the overall process. Next, we will describe how to install, setup, maintain, debug, and remove your BNU package. The `sysadm` interface is provided to help you through these procedures. Sections detailing the structure and format of the various BNU data base support, administrative, and uucp log files are then described. As you gain more experience with BNU administration, these sections can be used as reference material for fine tuning your computer's configuration. Lastly, we discuss some points on how to configure your BNU package to work with direct links, telephone lines, and local area networks.

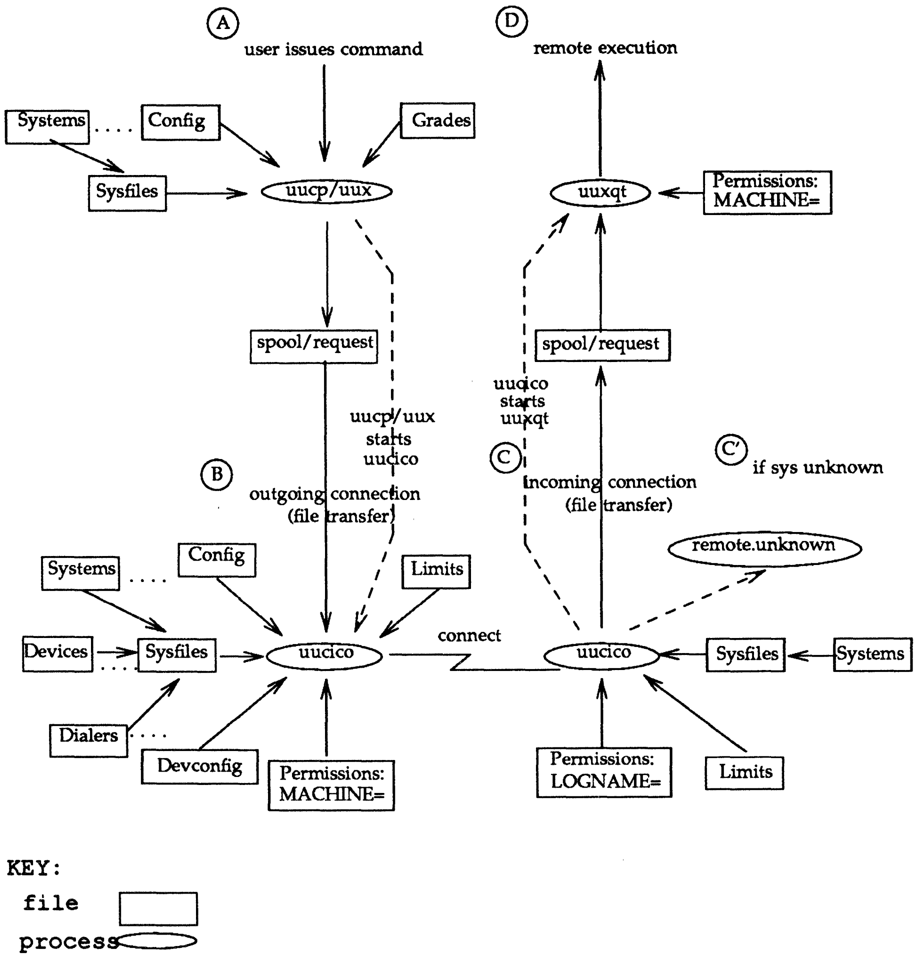
The Basic Networking Process

After you have properly installed and configured your BNU package, your computer will be ready to communicate with other computers that use the UNIX operating system. Here is a general view of how BNU accomplishes this communication process:

1. A user on your local machine issues a command requesting file transfer or remote execution communication with a remote computer (phase `A`). Several BNU data base support files are read to determine if the remote computer is accessible by your local computer and the priority of the user's request, as compared to other users' requests. This phase ends by queuing the user's request in a spool area on your local machine and triggering the next phase.
2. The `uucico` routine is triggered automatically (phase `B`). It reads several BNU data base support files to determine when the remote computer can be reached, how to establish the link to the remote computer, how to handle data flow between the local and remote computers, and if the maximum number of simultaneous requests for communication to the remote computer has been reached.
3. The `uucico` routine on the remote computer is triggered automatically when a call for communication is received from the local computer (phase `C`). In this phase, the remote `uucico` routine reads BNU data base support files on its computer to determine if the calling computer is allowed access, and what action to take if the calling computer is not allowed access (phase `C'`). For calling computers that are allowed access to the remote computer, the level of access is determined in this phase.
4. Requests initiated on the local computer may contain commands to be executed on the remote computer (phase `D`). When these commands arrive on the remote computer, they are stored in a spool area. The `uuxqt` routine on the remote computer runs these commands on the remote computer during this phase.

Here is a picture that describes the Basic Networking process:

Figure 7-6: Basic Networking Process



Networking Hardware

Before your computer can communicate with other computers, you must set up the hardware to complete the communications link. The cables and other hardware you need will depend on how you want to connect the computers: direct links, telephone lines, or local area networks.

Networking Programs

There are two categories of Basic Networking programs: user programs and administrative programs.

User Programs

You will find the user programs for Basic Networking in `/usr/bin`. No special permission is needed to use these programs; they are all fully described in the *User's Reference Manual*: `cu`, `ct`, `uucp`, `uuto`, `uupick`, `uux`, `uustat`, `uulog`, `uuglist`, `uuname`, `uudecode`, `uuencode`.

Administrative Programs

You will find most of the administrative programs in `/usr/lib/uucp`, along with Basic Networking shell scripts. The only exception is `uulog`, which is in `/usr/bin`. All of these commands are fully described in the *System Administrator's Reference Manual*: `uucleanup`, `Uutry`, `uucheck`.

You should use the `uucp` login ID only when you administer BNU because it owns the Basic Networking and spooled data files. The home directory of the `uucp` login ID is `/usr/lib/uucp`. (The other Basic Networking login ID is `nuucp`, used by remote computers to access your computer. Calls from `nuucp` are answered by `uucico`.)

Networking Daemons

There are three daemons in BNU. A daemon is a routine that runs as a background process and performs a system-wide public function. These daemons handle file transfers and command executions. They can also be run manually from the shell.

`uucico`	Selects the device used for the link, establishes the link to the remote computer, performs the required login sequence and permission checks, transfers files (if requested), logs results, and 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.
	uucico is executed by the uucp, uuto, and uux programs, after all the required files have been created, to contact the remote computer. It is also executed by uusched and Uutry.
uuxqt	Executes remote execution requests. It searches the spool directory for execute files (always named X.*file*) that have been sent from a remote computer. When an execute 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. If the files are present and can be accessed, uuxqt checks the Permissions file to verify that it has permission to execute the requested command. uuxqt is executed by the uudemon.hour shell script, which is started by cron.
uusched	Schedules the queued work in the spool directory. Before starting uucico, uusched randomizes the order in which remote computers will be called. uusched is executed by a shell script called uudemon.hour, which is started by cron.

Networking Support Files

There are three types of BNU support files:

Data Base	These files are responsible for much of the actual networking activity associated with your BNU package. They are located in /etc/uucp. In general, they determine: what computers your computer will communicate with, the devices over which the communication will take place, and the protocols for communicating with remote computers. See the section entitled "Data Base Support Files."
Administrative	Administrative files are created by network processes in spool directories to lock devices, hold temporary

	data, or store information about remote transfers or command executions. See the section entitled "Administrative Support Files."
Log	Log files keep track of overall statistics of your computer network, particularly in the areas of security and accounting. See the section entitled "Logs."

Basic Procedures

The following steps are performed to administer BNU.

Installation	Installing the BNU software package (placing it on your computer's hard disk) is first. To install your BNU package, you should follow the guidelines described under "Installing Software Packages" in the "Software Management" chapter of this guide.
Setup	Setting up the basic networking facility and configuring basic networking files. See the following section, "Setup Basic Networking Files."
Maintenance	Automatically and manually maintaining your basic networking files and transactions so that your operations continue to run smoothly. See "Basic Networking Maintenance" in this section.
Debugging	Identifying and correcting common problems in basic networking operations and administration. See "Basic Networking Debugging" in this section.
Removal	Removing the BNU software package (deleting it from your computer's hard disk) is last. To remove your BNU package, you should follow the guidelines described under "Removing Packages" in the "Software Management" chapter of this guide.

NOTE See "BNU Software Removal Considerations" in this section for information to consider before you remove your BNU software package.

Setup Basic Networking Files

The BNU *Setup* procedure provides the steps for initializing your BNU data base. This information allows your local and remote systems to communicate with each other. This procedure is done by using the `sysadm` subcommands and your favorite text editor.

Begin by typing `sysadm basic_networking` and selecting `setup`. This will give you the starting menu. From this point, you can begin setting up (initializing) the various data base files. Continue making interactive menu selections until you complete the task(s). To get help along the way, press the HELP function key. This will give you a detailed description of a menu selection. Pressing the CANCEL function key exits help mode.

Setting up the `Permissions`, `Devconfig`, `Sysfiles` and `Limits` files, and adding uucp logins are principal functions in the initial Basic Networking setup process. Here is more detailed information in these areas.

Set Up Permissions File

The default `/etc/uucp/Permissions` file provides the maximum amount of security for your computer. The file, as delivered, contains the following entry:

```
LOGNAME=nuucp
```

You can set additional parameters for each machine you communicate with:

- the ways it can receive files from your machine
- the directories it can read and write
- the commands it can execute remotely

See the section "Data Base Support Files" for information on how to set up this file. You can modify this file to include the entries you desire.

Set Up Devconfig File

The /etc/uucp/Devconfig file is only used if you are using BNU over an AT&T STARLAN NETWORK or some other Streams-based provider. If you are using an AT&T STARLAN NETWORK, the two entries shown in the Devconfig file are all you need in this file.

```
service=cu      device=STARLAN push=ntty:tirdwr
service=uucico  device=STARLAN push=ntty:tirdwr
```

You must also create an entry for STARLAN in your Devices file. Descriptions in the Devices file tell how to define Transport Interface devices.

Devconfig entries define the STREAMS modules that are used for a particular device. (The push= variable shows the modules and the order they are pushed on to a stream.) Different modules and devices can be defined for cu and uucico services. You can modify this file to include the entries you desire.

Set Up Sysfiles File

/etc/uucp/Sysfiles lets you assign different files to be used by uucp and cu as Systems, Devices, and Dialers files. Here are some cases where this optional file may be useful:

- You may want different Systems files so requests for cu login services can be made to addresses other than uucp services.
- You may want different Dialers files to use different chat scripts for cu and uucp.
- You may want to have multiple Systems, Dialers, and Devices files. The Systems file in particular may become large, making it convenient to split it into several smaller files.

The format of the Sysfiles file is described in the section entitled "Data Base Support Files." The following is an example of the file.

```
service=uucico  systems=Systems.cico:Systems\
                dialers=Dialers.cico:Dialers\
                devices=Devices.cico:Devices
service=cu      systems=Systems.cu:Systems\
                dialers=Dialers.cu:Dialers\
                devices=Devices.cu:Devices
```

You can modify this file to include the entries you desire.

Set Up `Limits` File

The `/etc/uucp/Limits` file is used to limit the maximum number of simultaneous `uucicos`, `uuxqts`, and `uuscheds` that are running on your machine.

See the section entitled "Data Base Support Files" for a format description of the `Limits` file. The following is an example of the file.

```
service=uucico max=5
service=uuxqt max=5
service=uusched max=2
```

You can modify this file to include the entries you desire.

Add `uucp` logins

You may add one or more administrative logins to your system so incoming `uucp` (`uucico`) requests from different remote machines can be handled differently. Each remote machine should have an entry in its `Systems` file for your machine that contains the login ID and password that you add to your `/etc/passwd` file.

The default in the `/etc/passwd` file is shown below.

```
uucp:x:5:5:0000-uucp(0000):/usr/lib/uucp
nuucp:x:10:10:0000-uucp(0000):/var/spool/uucppublic:/usr/lib/uucp/uucico
```

This entry shows that a login request by nuucp is answered by `/usr/lib/uucp/uucico`. The home directory is `/var/spool/uucppublic`. The *x* indicates that the encrypted password is stored in `/etc/shadow`.

Basic Networking Maintenance

BNU maintenance involves automatically and manually maintaining your basic networking files so that they don't become large and consume too much disk space. It also means keeping BNU running smoothly.

BNU comes with four shell scripts:

```
uudemon.poll
uudemon.hour
uudemon.admin
uudemon.cleanup
```

These scripts will poll remote machines, reschedule transmissions, and clean up old log files and unsuccessful transmissions. They should be used regularly to keep your basic networking running smoothly. Normally, they are run automatically with `cron(1M)` although they can also be run manually.

Automated Networking Maintenance (`cron`)

BNU is delivered with entries for shell scripts in the `/var/spool/cron/crontabs/root` file. These entries will automatically handle some BNU administrative tasks for you. The shell scripts are in `/usr/lib/uucp`.

In multi-user mode, tasks scheduled by the `crontab` command are automatically performed by cron. The `crontab` file for root is generally used to schedule regular BNU maintenance using the following shell scripts:

`uudemon.poll`

The `uudemon.poll` shell script, as delivered, does the following:

- Reads the `Poll` file (`/etc/uucp/Poll`).
- If any of the machines in the `Poll` file are scheduled to be polled, a work file (`C.sysnxxxx`) is placed in the `/var/spool/uucp/`*nodename* directory, where *nodename* is replaced by the name of the machine being polled.

By default, the shell script is scheduled to run twice an hour, just before `uudemon.hour`, so that the work files will be there when `uudemon.hour` is called. The default root crontab entry for `uudemon.poll` is as follows:

```
1,30 * * * * /usr/lib/uucp/uudemon.poll > /dev/null
```

`uudemon.hour`

The `uudemon.hour` shell script you receive with your machine does the following:

- Calls the `uusched` program to search the spool directories for work files (C.) that have not been processed and schedules these files for transfer to a remote machine.
- Calls the `uuxqt` daemon to search the spool directories for execute files (X.) that have been transferred to your computer and were not processed at the time they were transferred.

The default root crontab entry for `uudemon.hour` is as follows:

```
41,11 * * * * /usr/lib/uucp/uudemon.hour > /dev/null
```

As delivered, this is run twice an hour. You may want it to run more often if you expect high failure rates.

`uudemon.admin`

The `uudemon.admin` shell script, as delivered, does the following:

- Runs the `uustat` command with -p and -q options. The -q reports on the status of work files (C.), data files (D.), and execute files (X.) that are queued. The -p prints process information for networking processes listed in the lock files (`/var/spool/locks`).

- Sends resulting status information to the uucp administrative login via `mail(1)`.

There is no default crontab entry for `uudemon.admin`. The following is recommended:

```
48 8,12,16 * * * /bin/su uucp -c "/etc/uucp/uudemon.admin" > /dev/null
```

`uudemon.cleanup`

The delivered `uudemon.cleanup` shell script does the following:

- Takes log files for individual machines from the `/var/spool/uucp/.Log` directory, merges them, and places them in the `/var/spool/uucp/.Old` directory with other old log information. If log files get large, the `ulimit` parameter may need to be increased.
- Removes work files (C.) 7-days old or older, data files (D.) 7 days old or older, and execute files (X.) two days old or older from the spool directory
- Returns mail that cannot be delivered to the sender
- Mails a summary of the status information gathered during the current day to the uucp administrative login

There is no default crontab entry for `uudemon.cleanup`. The following is recommended:

```
45 23 * * * ulimit 5000; /usr/bin/su uucp -c \
"/usr/lib/uucp/uudemon.cleanup" \
 > /dev/null 2>&1
```

`uudemon.cleanup` is described in detail below, under "Cleaning up Log Files."

Manual Maintenance

Some files may grow indirectly from uucp and other Basic Networking activities. Here are two files you should check and delete if they have become too large.

`/usr/adm/sulog`	This file keeps a history of all super user commands. Since the uudemon entries in the `/usr/cron/root` file use the `su` command, the `sulog` will grow over time. After examining it for tampering, you should delete this file if it becomes too large.
`/usr/lib/cron/log`	This file is a log of `cron` activities. While it grows with use, it is automatically truncated when the system goes to the multi-user state.

Cleaning up Log Files

`uudemon.cleanup` is a shell script that should be invoked daily by `cron` to clean up uucp's spool directory and to consolidate and dispose of the log files that currently exist for uucp. Currently, `uudemon.cleanup` uses two techniques to clean up the log files:

1. The *Multi-day* (multi) technique. Three days of logs are kept in files called `Old-1`, `Old-2`, and `Old-3`. Whenever `uudemon.cleanup` is run, `Old-2` is renamed `Old-3`, `Old-1` is renamed `Old-2`, and the current log is renamed `Old-1` and stored in `/var/spool/uucp/.Old/Old-1`.
2. The *Single-day* (single) technique. The current log is moved to `/var/spool/uucp/.Old`. This means that the log is preserved for one day only (assuming that `uudemon.cleanup` is run daily).

Although `uudemon.cleanup` does take care of removing old log entries, as an administrator, you should monitor the size of the log files. The exact procedure `uudemon.cleanup` uses to remove old log entries varies according to the type of log file. Here's a description of how the `uucp` log files will be cleaned up.

Table 7-5: Summary of BNU Log Files

File Use	File Name (beginning with `/var/uucp`)	Cleanup Technique
Command	`.../.Admin/command`	single
History	`.../.Log/[uucp|uucico|uux|uuxqt]/system`	multi
Foreign	`.../.Admin/Foreign`	single
Error	`.../.Admin/errors`	single
Transfer	`.../.Admin/xferstats`	single
Accounting	`.../.Admin/account`	multi
Security	`.../.Admin/security`	multi
Performance	`.../.Admin/perflog`	single

Basic Networking Debugging

The *Debug* procedures are intended to help you identify and correct common problems in basic networking operations and administration. Here is a list of the available monitoring tools you can use to detect and solve basic networking problems:

```
uustat(1)
cu(1)
Uutry(1)
uuname(1M)
uulog(1)
uucheck(1)
```

Check Basic Information

There are several commands you can use to check for Basic Networking information.

`uuname`	Use this command to list those machines your machine can contact.
`uulog`	Use this command to display the contents of the log directories for particular hosts.

`uucheck -v`	Run this command to check for the presence of files and directories needed by uucp. This command also checks the `Permissions` file and provides information on the permissions you have set up.

Check for Faulty ACU/Modem

You can check if the automatic call units or modems are not working properly in several ways.

- Run `uustat -q`. This will give counts and reasons for contact failure.
- Run `cu -d -l`*line*. This will let you call over a particular communications line and print debugging information on the attempt. If the communications line, *line*, is connected to an autodialer, you must add a telephone number at the end of the command line you execute. Otherwise, *line* must be defined as `direct` in the `Devices` file.

Check Systems File

Check that you have up-to-date information in your systems file if you are having trouble contacting a particular machine. Some things that may be out of date for a machine are its:

- phone number
- login
- password

Debug Transmissions

If you are unable to contact a particular machine, you can check out communications to that machine with `Uutry` and uucp.

Step 1: Simply to try to make contact, run:

```
$ /usr/lib/uucp/Uutry -r machine
```

where *machine* is replaced with the node name of the machine you are having problems contacting. This command will:

1. Start the transfer daemon (`uucico`) with debugging. You will get more debugging information if you are `root`.
2. Direct the debugging output to `/tmp/`*machine*,
3. Print the debugging output to your terminal, using `tail -f`. Hit BREAK to end output.

You can copy the output from `/tmp/`*machine* if you want to save it.

Step 2: If `Uutry` doesn't isolate the problem, try to queue a job by running:

```
$ uucp -r file machine!/dir/file
```

where *file* is replaced by the file you want to transfer, *machine* is replaced by the machine you want to copy to, and *dir/file* is where the file will be placed on the other machine. The `-r` option will queue a job but not start the transfer.

Now use `Uutry` again. If you still cannot solve the problem, you may need to call support personnel. Save the debugging output; it will help diagnose the problem.

BNU Software Removal Considerations

Since removing BNU leaves you without a means of communicating with other computers, this should be a last resort for freeing disk space. Before you remove the BNU software, you may want to save the information in `/etc/uucp`.

Data Base Support Files

The BNU support files are in the `/etc/uucp` directory. Most changes to these files can be made using the System Administration Menu commands described in the *Setup* procedure. The descriptions below, however, provide details on the structure of these files so you can edit them manually.

`Config`	Contains a list of variable parameters within BNU. The administrator can set these parameters to configure the network manually.

`Devconfig`	This file is used to configure your network connections on STARLAN or some other network provider.
`Devices`	Contains information concerning the location and line speed of automatic call units, direct links, and network devices.
`Dialcodes`	This file contains dial-code abbreviations that may be used in the *telephone number* field of `Systems` file entries.
`Dialers`	Contains character strings required to communicate with network devices, automatic calling units, and direct links.
`Grades`	This file is used to define the job grades, and the permissions associated with each job grade, that users may specify to queue jobs to a remote computer.
`Limits`	This file defines the maximum number of simultaneous `uucico`s, `uuxqt`s, and `uusched`s permitted on your machine.
`Permissions`	This file defines the level of access that is granted to computers when they attempt to transfer files or execute remote commands on your computer.
`Poll`	This file defines computers that are to be polled by your system and when they are polled.
`Sysfiles`	This file is used to assign different or multiple files to be used by `uucico` and `cu` as `Systems`, `Devices`, and `Dialers` files.
`Systems`	Contains information needed by the `uucico` daemon and the `cu` program to establish a link to a remote computer, such as the name of the remote computer, the name of the connecting device associated with the remote computer, when the computer can be reached, telephone number or network address, login ID, and password.

There is one file that may be considered part of the supporting data base, but is not directly related to the process of establishing a link and transferring files. This file—`remote.unknown`—is described briefly in the section entitled "Other Networking Files."

Config File

The /etc/uucp/Config file allows the administrator to override certain parameters within BNU manually. Each entry in the Config file has the following format:

parameter=value

Where *parameter* is one of the configurable parameters and *value* is the value to be assigned to that parameter. See the Config file provided with your system for a complete list of configurable parameter names.

The following Config file entry sets the default protocol ordering to "Gge" and changes the "G" protocol defaults to 7 windows and 512 byte packets.

```
Protocol=G(7,512)ge
```

Devices File

The Devices file (/etc/uucp/Devices) contains information for all the devices that may be used to establish a link to a remote computer. Provisions are made for several types of devices, such as automatic call units, direct links, and network connections.

NOTE 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 Line Line2 Class Dialer-Token-Pairs

These fields are defined as:

Type This field may contain one of two keywords (Direct or ACU), the name of a Local Area Network, switch, or system.

Direct	This keyword indicates a Direct Link to another computer or a switch.

`ACU`	This keyword specifies that the link to a remote computer is made through an automatic call unit (Automatic Dial Modem). This modem may be connected either directly to your computer or indirectly through a Local Area Network (LAN) switch.
LAN_Switch	This is the name of the LAN or switch. For instance, `STARLAN` could be the name for the AT&T STARLAN network. `Develcon` could be the name for a Develcon switch connection.
Sys-Name	This value specifies a direct link to a particular computer. (*Sys-Name* is replaced by the name of the computer.) This naming scheme is used to convey the fact that the line associated with this `Devices` entry is for a particular computer in the `Systems` file.

The keyword used in the *Type* field is matched against the third field of `Systems` file entries as shown below:

```
Devices: ACU term/11 - 1200 penril

Systems: eagle Any ACU 1200 3251 ogin: nuucp \
             ssword: Oakgrass
```

You can designate a protocol to use for a device within this field. See the "Protocols" section at the end of the description of this file.

Line This field contains the device name of the line (port) associated with the `Devices` entry. For instance, if the Automatic Dial Modem for a particular entry was attached to the `/dev/term/11` line, the name entered in this field would be `term/11`. There is an optional modem control flag, *M*, that can be used in the *Line* field to indicate that the device should be opened without waiting for a carrier. For example:

```
term/11,M
```

Line2 If the keyword `ACU` was used in the *Type* field and the ACU is an 801 type dialer, *Line2* would contain the device name of the 801 dialer. (801 type ACUs do not contain a modem. Therefore, a separate modem is required and would be connected to a different line, defined in the *Line* field.) This means that one line would be allocated to the modem and another to the dialer. Since non-801 dialers will not normally use this configuration, the *Line2* field will be ignored by them, but it must still contain a hyphen (-) as a placeholder.

Class If the keyword `ACU` or `Direct` is used in the *Type* field, *Class* may be just the speed of the device. However, it may 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. In such a case, it becomes necessary to distinguish which line(s) should be used for internal communications and which should be used for external communications. The keyword used in the *Class* field of the `Devices` file is matched against the fourth field of `Systems` file entries as shown below:

```
Devices: ACU ttyll - D1200 penril

Systems: eagle Any ACU D1200 3251 ogin: nuucp \
         ssword: Oakgrass
```

Some devices can be used at any speed, so the keyword `Any` may be used in the *Class* field. If `Any` is used, the line will match any speed requested in a `Systems` file entry. If this field is `Any` and the `Systems` file *Class* field is `Any`, the speed defaults to 1200 bps.

Dialer-Token-Pairs:

This field contains pairs of dialers and tokens. The *dialer* portion may be the name of an automatic dial modem, a LAN switch, or it may be `direct` or `uudirect` for a Direct Link device. You can have any number of Dialer-Token-Pairs. The *token* portion may be supplied immediately following the *dialer* portion, or, if not present, it will be taken from a related entry in the `Systems` file.

This field has the format:

dialer token [*dialer token*]

where the last pair may or may not be present, depending on the associated device (dialer). In most cases, the last pair contains only a *dialer* portion and the *token* portion is retrieved from the *Phone* field of the `Systems` file entry.

A valid entry in the *dialer* portion may be defined in the `Dialers` file or may be one of several special dialer types. These special dialer types are compiled into the software and are therefore available without having entries in the `Dialers` file.

801	Bell 801 auto dialer
TLI	Transport Level Interface Network (without STREAMS)
TLIS	Transport Level Interface Network (with STREAMS)

The *Dialer-Token-Pairs* (*DTP*) field may be structured differently, depending on the device associated with the entry:

- If an automatic dialing modem is connected directly to a port on your computer, the *DTP* field of the associated `Devices` file entry will only have one pair. This pair would normally be the name of the modem. This name is used to match the particular `Devices` file entry with an entry in the `Dialers` file. Therefore, the *dialer* field must match the first field of a `Dialers` file entry as shown below:

  ```
  Devices: ACU term/11 - 1200 att2212c
  Dialers:  att2212c =+-, "" atzod,o12=y,o4=n\r\c \
          \006 atT\T\r\c ed
  ```

  Notice that only the *dialer* portion (`att2212c`) is present in the *DTP* 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; backslash sequences are described below.)

- If a direct link is established to a particular computer, the *DTP* field of the associated entry would contain the keyword `direct` or `uudirect`. This is true for both types of direct link entries, `Direct` and *System-Name* (refer to discussion on the *Type* field).

- If you wish to communicate with a computer that is on the same local network switch as your computer, your computer must first access the switch and the switch can make the connection to the other computer. In this type of entry, there is only one pair. The *dialer* portion is used to match a `Dialers` file entry as shown below:

```
Devices: develcon term/13 - 1200 develcon
Dialers: develcon "" "" \pr\ps\c est:\007 \E\D\e \007
```

As shown, the *token* portion is left blank. This indicates that it is retrieved from the `Systems` file. The `Systems` file entry for this particular computer will contain the token in the *Phone* field, which is normally reserved for the telephone number of the computer (refer to `Systems` file, *Phone* field). This type of *DTP* contains an escape character (`\D`), which ensures that the contents of the *Phone* field will not be interpreted as a valid entry in the `Dialcodes` file.

- If an automatic dialing modem is connected to a switch, your computer must first access the switch and the switch will make the connection to the automatic dialing modem. This type of entry requires two *dialer-token-pairs*. The *dialer* portion of each pair (fifth and seventh fields of entry) will be used to match entries in the `Dialers` file as shown below:

```
Devices:  ACU term/14 - 1200 develcon dial att2212c

Dialers:  develcon "" "" \pr\ps\c est:\007 \E\D\e \007
Dialers:   att2212c =+-, "" atzod,o12=y,o4=n\r\c \006 atT\T\r\c ed
```

In the first pair, `develcon` is the dialer and `dial` is the token that is passed to the Develcon switch to tell it which device (auto dial modem) to connect to your computer. This token would be unique for each LAN switch since each switch may be set up differently. Once the modem has been connected, the second pair is accessed, where `att2212c` is the dialer and the token is retrieved from the `Systems` file.

There are two escape characters that may appear in a *DTP* field:

`\T` Specifies that the *Phone* (*token*) field should be translated using the `Dialcodes` file. This escape character is normally placed in the `Dialers` file for each caller script associated with an automatic dial modem. Therefore, the translation will not take place until the caller script is accessed.

`\D` Indicates that the *Phone* (*token*) field should not be translated using the `Dialcodes` file. If the dialer is an internal dialer, `\T` is the default. Otherwise, `\D` is the default. A `\D` is also used in the `Dialers` file with entries associated with network switches (develcon and micom).

Protocols

You can choose the protocol to use with each device. Usually, it is not needed since you can use the default. If you do specify the protocol, you must do so in the form *Type,Protocol[(parameters)]* (for example, `STARLAN,eg`). Available protocols are:

g This is a generic packet protocol. It provides error detection and retransmission intended for use over potentially noisy lines. By its nature, it is relatively slow. Two parameters characterize the `g` protocol, *windows* and *packetsize*. *windows* indicates the number of packets which may be transmitted without waiting for an acknowledgement from the remote host. *packetsize* indicates the number of data bytes in each packet. *windows* value is set at 7, and *packetsize* is set at 64 bytes.

G This protocol is identical to the `g` protocol in that it provides the same error detection and retransmission. However, in addition, the `G` protocol allows the number of windows and the packet size to be varied to match the characteristics of the transmission medium. When properly configured, performance can be significantly better than the `g` protocol. *windows* may range from 1 to 7, and *packetsize* may range from 32 to 4096 bytes, in powers of 2 (that is, 32, 64, 128, 256, 512, 1024, 2048, 4096).

e This protocol assumes error free transmission and performs no error checking or retransmission. Therefore it is the fastest of these protocols. It should be used for reliable local area networks. There are no parameters to be tuned within the `e` protocol.

Here is an example that uses the e protocol over a STARLAN local area network. If the e protocol is not available, g will be used.

```
STARLAN,eg starlan - - TLIS \D
```

Here is an example that uses the G protocol on a high speed modem. The number of windows is set to 7, and the packetsize is 512 bytes. If the G protocol is unavailable, the standard g protocol will be used.

```
ACU,G(7,512)g term/11 - 9600 att2296a
```

Presumably, seven windows with a packet size of 512 bytes will provide optimum throughput for the specified device.

For incoming connections, the preferred protocol priority and parameters may be specified in the `Config` file using the *Protocol* parameter.

Dialers File

The `Dialers` file (`/etc/uucp/Dialers`) specifies the initial conversation that must take place on a line before it can be made available for transferring data. This conversation is usually a sequence of character strings that is transmitted and expected, and it is often used to dial a telephone number using an Automatic Call Unit.

As shown in the above examples, the fifth and subsequent odd numbered fields in the `Devices` file is an index into the `Dialers` file or internal list of special dialer types (801, TLI, or TLIS). If the match succeeds, the `Dialers` entry is interpreted to perform the dialer conversation. Each entry in the `Dialers` file has the following format:

dialer substitutions expect-send ...

The *dialer* field matches the fifth and additional odd numbered fields in the `Devices` file. The *substitutions* field is a translate string: the first of each pair of characters is mapped to the second character in the pair. This is usually used to translate = and – into whatever the dialer requires for "wait for dialtone" and "pause."

The remaining *expect-send* fields are character strings. Figure 7-7 shows some character strings distributed with BNU in the `Dialers` file.

Figure 7-7: Sample Character Strings in `Dialers` File

```
penril =W-P "" \d > Q\c : \d- > s\p9\c )-W\p\r\ds\p9\c-)
                 y\c : \E\TP > 9\c OK
ventel =&-% "" \r\p\r\c $ <K\T%%\r>\c ONLINE!
vadic =K-K "" \005\p *-\005\p-*\005\p-* D\p BER? \E\T\e \r\c LINE
develcon "" "" \pr\ps\c est:\007 \E\D\e \n\007
micom "" "" \s\c NAME? \D\r\c GO
direct
uudirect "" "" \r\d in:--in:
rixon =&-% "" \r\r\d $ s9\c )-W\r\ds9\c-) s\c : \T\r\c $ 9\c LINE
hayes =,-, "" \dAT\r\c OK\r \EATDT\T\r\c CONNECT
att4000 =,-, "" ATZ\r\p\p  OK\r ATZ\r OK\r\c \EATDT\T\r\c CONNECT
att4024 =+-, "" atzod,o12=y,o4=n\r\c \006 atT\T\r\c ed
att2212c =+-, "" atzod,o12=y,o4=n\r\c \006 atT\T\r\c ed
att2224b =+-, "" atT\T\r\c ed
att2224ceo =+-, "" atzod,o12=y,o4=n,\\n3\\c1\\j0\\q0\\g0\r\c \006
          atT\T\r\c Connected
att2224g =+-, "" atzod,o12=y,o4=n,o1=n\r\c \006
          atz\\n3\\c1\\j0\\q0\\g0\r\c "" \datT\T\r\c Connected
att2224 =+-,"" \r\c :--: T\T\r\c red
att2248a =+-, "" atzod,o12=y\r\c \006 atT\T\r\c Connected
att2296a =+-, "" atzod,o12=y,o50=y,o51=n,o55=n,o69=n\r\c \006
          atz\\n3\\c1\\j0\\q0\\g0\r\c "" \datT\T\r\c Connected
nls "" "" NLPS:000:001:1\N\c
```

The meaning of some of the escape characters (those beginning with "\") used in the `Dialers` file are listed below:

`\p`	Pause (approximately ¼ to ½ second)
`\d`	Delay (approximately 2 seconds)
`\D`	Telephone number or token without `Dialcodes` translation
`\T`	Telephone number or token with `Dialcodes` translation
`\K`	Insert a BREAK
`\E`	Enable echo checking (for slow devices)
`\e`	Disable echo checking

`\r`	Carriage return
`\c`	No new-line or carriage return
`\M`	Turn on `CLOCAL` flag
`\m`	Turn off `CLOCAL` flag
`\n`	Send new-line
`\`*nnn*	Send character represented by octal number *nnn*

Additional escape characters that may be used are listed in the section "`Systems` File."

The att2212c entry in the `Dialers` file is executed as follows. First, the telephone number argument is translated, replacing any = with a `W` (wait for dialtone) and replacing any – with a `P` (pause). The handshake given by the remainder of the line works as follows:

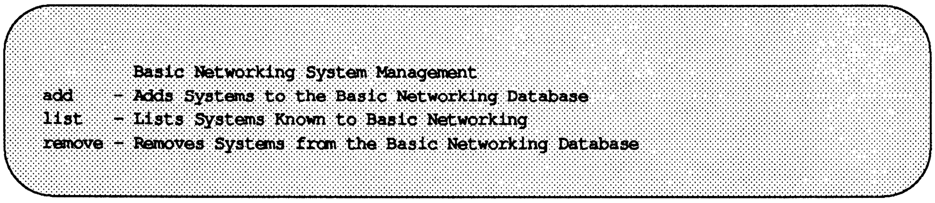
`""`	Wait for nothing. (In other words, proceed to the *expect-send* string.)
`=+-`	Secondary dial tone and pause.
`atzod`	Enter command mode, reset modem, set options to default.
`o12=y`	Set option 12 to 'y' (transparent data mode).
`o4=n\r\c`	Set option 4 to 'n' (don't disconnect on received spaces) terminate with a carriage return but no newline.
`\006`	Wait for acknowledge signal (ACK).
`atT\T\r\c`	Enter command mode. Use tone dialing. Translate the phone number and terminate with a carriage return, but no newline.
`ed`	Expect "`ed`" (answered).

Systems File

The Systems file (/etc/uucp/Systems) contains the information needed by the uucico daemon to establish a communication link to a remote computer. Each entry in the file represents a computer that can be called by your computer. In addition, the Basic Networking software is configured to prevent any computer that does not appear in this file from logging in on your computer (refer to the section "Data Base Support Files" in this chapter for a description of the remote.unknown file). More than one entry may be present for a particular computer. The additional entries represent alternative communication paths that will be tried in sequential order.

The screen below allows you to do BNU systems management. It can be brought up on the screen by typing sysadm systems.

Figure 7-8: Basic Networking System Management Menu



```
            Basic Networking System Management
  add    - Adds Systems to the Basic Networking Database
  list   - Lists Systems Known to Basic Networking
  remove - Removes Systems from the Basic Networking Database
```

If you want to bypass the menu system, you may issue equivalent commands directly to the shell as shown in Table 7-6:

Table 7-6: Shell Commands for Basic Networking System Management

Task Description	Menu Item	Shell Command
Add systems to the Basic Networking Database	`add`	`vi /etc/uucp/systems`
List systems in the Basic Networking Database	`list`	`cat /etc/uucp/systems`

Table 7-6: Shell Commands for Basic Networking System Management (continued)

Task Description	Menu Item	Shell Command
Remove systems from the Basic Networking Database	`remove`	`vi /etc/uucp/systems`

Using the `Sysfiles` file, you can define several files to be used as "Systems" files. See the description of the `Sysfiles` file for details. Each entry in the `Systems` file has the following format:

System-Name Time Type Class Phone Login

These fields are defined as:

System-name
: This field contains the node name of the remote computer.

Time
: This field is a string that specifies the day-of-week and time-of-day when the remote computer can be called. The format of the *Time* field is:

daytime[*;retry*]

The day portion may be a list containing some of the following:

`Su Mo Tu We Th Fr Sa`	for individual days
`Wk`	for any week-day (Mo Tu We Th Fr)
`Any`	for any day
`Never`	for a passive arrangement with the remote computer. If the *Time* field is `Never`, your computer will never initiate a call to the remote computer. 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 the section "`Permissions` File").

The *time* portion should be a range of times specified in 24-hour notation, for example 0800-1230 for "8:30 a.m. to 12:30 p.m." If no *time* portion 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 a.m. and 8 a.m. An optional subfield, *retry*, is available to specify the minimum time (in minutes) before a retry, following a failed attempt. The default wait is 60 minutes. The subfield separator is a semicolon (;). For example, `Any;9` is interpreted as call any time, but wait at least 9 minutes before retrying after a failure occurs. Here is an example:

```
Wk 1700-0800,Sa,Su
```

This example allows calls from 5:00 p.m. to 8:00 am, Monday through Friday, and calls any time Saturday and Sunday. The example would be an effective way to call only when telephone rates are low, if immediate transfer is not critical.

Type This field contains the device type that should be used to establish the communication link to the remote computer. The keyword used in this field is matched against the first field of `Devices` file entries as shown below:

```
Systems:   eagle Any ACU,g D1200 3251 ogin: nuucp \
            ssword: Oakgrass

Devices:   ACU tty11 - D1200 penril
```

You can define the protocol used to contact the system by adding it on to the *Type* field. The example above shows how to attach the protocol `g` to the device type `ACU`. See the information in the section "Protocols" under "`Devices` File" for details.

Class This field specifies the transfer speed of the device used in establishing the communication link. It may contain a letter and speed (for example, C1200, D1200) to differentiate between classes of dialers (refer to the discussion of the *Class* field under "`Devices` File"). Some devices can be used at any speed, so the keyword `Any` may be used. This field must match the *Class* field in the associated `Devices` file entry as shown below:

```
Systems:  eagle Any ACU D1200 NY3251 ogin: nuucp \
           ssword: Oakgrass

Devices:  ACU tty11 - D1200 penril
```

If information is not required for this field, use a – as a place holder for the field.

Phone This field allows you to specify the telephone number (token) of the remote computer for automatic dialers (LAN switches). The telephone 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. For example:

```
Systems:  eagle Any ACU D1200 NY3251 ogin: nuucp \
           ssword: Oakgrass

Dialcodes:  NY 9=1212555
```

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 4 seconds before dialing the next digit.

If your computer is connected to a LAN switch, you may access other computers that are connected to that switch. The `Systems` file entries for these computers will not have a telephone number in the *Phone* field. Instead, this field will contain the token that must be passed on to the switch so it will know 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 ensure that this field is not translated using the `Dialcodes` file.

Login This field contains login information given as a series of fields and subfields of the 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 may be made up of subfields of the form:

expect[*-send-expect*]...

where the *send* is sent if the prior *expect* is not successfully read and the *expect* following the *send* is the next expected string. For example, with login--login, uucp will expect login. If uucp gets login, it will go on to the next field. If it does not get login, it will send nothing followed by a new line, then look for login again. If no characters are initially expected from the remote computer, the characters "" (null string) should be used in the first *expect* field. Note that all *send* fields will be sent followed by a new-line unless the *send* string is terminated with a \c.

Here is an example of a Systems file entry that uses an expect-send string:

```
owl Any ACU 1200 Chicago6013 "" \r ogin:-BREAK-ogin: \
uucpx word: xyzzy
```

This example says don't wait, just send a carriage return and wait for ogin: (for Login:). If you don't get ogin, send a BREAK. When you do get ogin: send the login name uucpx, then when you get word: (for Password:), send the password xyzzy.

There are several escape characters that cause specific actions when they are a part of a string sent during the login sequence. The following escape characters are useful when using BNU communications:

\N	Send or expect a null character (ASCII NUL)
\b	Send or expect a backspace character
\c	If at the end of a string, suppress the new-line that is normally sent. Ignored otherwise.
\d	Delay two seconds before sending or reading more characters
\p	Pause for approximately ¼ to ½ second
\E	Start echo checking. (From this point on, whenever a character is transmitted, it will wait for the character to be received before doing anything else.)
\e	Echo check off

`\M`	Turn on `CLOCAL` flag
`\m`	Turn off `CLOCAL` flag
`\n`	Send a new-line character
`\r`	Send or expect a carriage-return
`\s`	Send or expect a space character
`\t`	Send or expect a tab character
`\\`	Send or expect a \ character
`EOT`	Send or expect EOT new-line twice
`BREAK`	Send or expect a break character
`\K`	Same as BREAK
`\ddd`	Collapse the octal digits (ddd) into a single character

`Dialcodes` File

The `Dialcodes` file (`/etc/uucp/Dialcodes`) contains the dial-code abbreviations that can be used in the *Phone* field of the `Systems` file. Each entry has the format:

abb dial-seq

where *abb* is the abbreviation used in the `Systems` file *Phone* field and *dial-seq* is the dial sequence that is passed to the dialer when that particular `Systems` file entry is accessed.

The entry

```
jt 9=555-
```

would be set up to work with a *Phone* field in the `Systems` file such as `jt7867`. When the entry containing `jt7867` is encountered, the sequence 9=555-7867 would be sent to the dialer if the token in the dialer-token-pair is `\T`.

`Permissions` File

The `Permissions` file (`/etc/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. Another option is available that specifies the commands that a remote site can execute on the local computer.

The following items should be considered when using the `Permissions` file to restrict the level of access granted to remote computers:

- All login IDs used by remote computers to login for uucp communications must appear in one and only one `LOGNAME` entry.
- Any site that is called whose name does not appear in a `MACHINE` entry, will have the following default permissions/restrictions:
    1. Local send and receive requests will be executed.
    2. The remote computer can send files to your computer's `/var/spool/uucppublic` directory.
    3. The commands sent by the remote computer for execution on your computer must be one of the default commands; usually `rmail`.
- When a remote machine calls you, unless you have a unique login and password for that machine, you don't know if the machine is who it says it is.

How Entries Are Structured

Each entry is a logical line with physical lines terminated by a backslash (\) to specify that the entry continues on the next line. Entries are made up of options delimited by white space. Each option is a name/value pair in the following format:

name=value

Note that no white space is allowed within an option assignment.

Comment lines begin with a pound 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 logs in on (calls) your computer.
`MACHINE`	Specifies permissions that take effect when your computer logs in on (calls) a remote computer.

Options

This section describes each option, specifies how they are used, and lists their default values.

`REQUEST`

When a remote computer calls your computer and requests the transfer of a file, this request is granted or denied based on the value of the `REQUEST` option. The string

```
REQUEST=yes
```

specifies that the remote computer can request the transfer of files from your computer. The string

```
REQUEST=no
```

specifies that the remote computer cannot request file transfers from your computer. This is the default value. It will be 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`

When a remote computer calls your computer and completes its work, it may attempt to take work your computer has queued for it. The `SENDFILES` option specifies whether your computer can send the work queued for the remote computer.

The string

```
SENDFILES=yes
```

specifies that your computer may send the work that is queued for the remote computer as long as it logged in as one of the names in the `LOGNAME` option. This string is mandatory if your computer is in a "passive mode" with respect to the remote computer.

The string

```
SENDFILES=call
```

specifies that files queued in your computer will be sent only when your computer calls the remote computer. The call value is the default for the `SENDFILES` option. This option is only significant in `LOGNAME` entries, since `MACHINE` entries apply when calls are made out to remote computers. If the option is used with a `MACHINE` entry, it will be ignored.

`READ` and `WRITE`

These options specify the various parts of the file system 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=/var/spool/uucppublic
WRITE=/var/spool/uucppublic
```

The strings

```
READ=/ WRITE=/
```

specify permission to access any file that can be accessed by a local user whose access permissions are set to "other."

The value of these entries is a colon separated list of path names. The `READ` option is for requesting files, and the `WRITE` option for depositing files. One of the values must be a component of any full path name of a file coming in or going out. To grant permission to deposit files in `/usr/news` as well as the public directory, the following values would be used with the `WRITE` option:

```
WRITE=/var/spool/uucppublic:/usr/news
```

It should be pointed out that if the `READ` and `WRITE` options are used, all path names must be specified because the path names are not added to the default list. For instance, if the

/usr/news path name was the only one specified in a WRITE option, permission to deposit files in the public directory would be denied.

You should be careful what directories you make accessible for reading and writing by remote systems. For example, you probably wouldn't want remote computers to be able to write over your /etc/passwd file, so /etc shouldn't be open to writes.

NOREAD and NOWRITE

The NOREAD and NOWRITE options specify exceptions to the READ and WRITE options or defaults. The strings

```
READ=/ NOREAD=/etc WRITE=/var/spool/uucppublic
```

would permit reading any file except those in the /etc directory (and its subdirectories—remember, these are prefixes) and writing only to the default /var/spool/uucppublic directory. NOWRITE works in the same manner as the NOREAD option. The NOREAD and NOWRITE can be used in both LOGNAME and MACHINE entries.

CALLBACK

The CALLBACK option is used in LOGNAME entries to specify that no transaction will take 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 fairly certain it is the machine it says it is. If you are doing long data transmissions, you can choose the machine that will be billed for the longer call.

The string

```
CALLBACK=yes
```

specifies that your computer must call the remote computer back before any file transfers will take place.

The default for the COMMAND option is

```
CALLBACK=no
```

The CALLBACK option is very rarely used. Note that if two sites have this option set for each other, a conversation will never get started.

COMMANDS

CAUTION The COMMANDS option can compromise the security of your system. Use it with extreme care.

The uux program will generate remote execution requests and queue them to be transferred to the remote computer. Files and a command are sent to the target computer for remote execution. The COMMANDS option can be used in MACHINE entries to specify the commands that a remote computer can execute on your computer. Note that COMMANDS is not relevant in a LOGNAME entry; COMMANDS in MACHINE entries define command permissions whether we call the remote system or it calls us.

The string

```
COMMANDS=rmail
```

specifies the default commands that a remote computer can execute on your computer. If a command string is used in a MACHINE entry, the default commands are overridden. For instance, the entry

```
MACHINE=owl:raven:hawk:dove \
COMMANDS=rmail:rnews:lp
```

overrides the COMMANDS default so that the computers owl, raven, hawk, and dove can now execute rmail, rnews, and lp on your computer.

NOTE See the section "Permissions File" in this chapter.

In addition to the names as specified above, there can be full path names of commands. For example,

```
    COMMANDS=rmail:/usr/lbin/rnews:/usr/local/lp
```

specifies that the command `rmail` uses the default path. The default path for remote execution is `/usr/bin`. When the remote computer specifies `rnews` or `/usr/lbin/rnews` for the command to be executed, `/usr/lbin/rnews` will be executed regardless of the default path. Likewise, `/usr/local/lp` is the `lp` command that will be executed.

CAUTION Including the `ALL` value in the list means that any command from the remote computer(s) specified in the entry will be executed. If you use this value, you give the remote computer full access to your computer. Be careful. This allows far more access than normal users have.

The string

```
COMMANDS=/usr/lbin/rnews:ALL:/usr/local/lp
```

illustrates two points:

- The `ALL` value can appear anywhere in the string.
- The path names specified for `rnews` and `lp` will be used (instead of the default) if the requested command does not contain the full path names for `rnews` or `lp`.

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` The `VALIDATE` option is used in conjunction with the `COMMANDS` option when specifying commands that are potentially dangerous to your computer's security. It is used to provide a certain degree of verification of the caller's identity. The use of the `VALIDATE` option requires that privileged computers have a unique login/password for uucp transactions. An important aspect of this validation is that the login/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 on top of 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 users on the remote computer, do not provide that computer with a privileged login and password.

The LOGNAME entry

```
    LOGNAME=uucpfriend VALIDATE=eagle:owl:hawk
```

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. If an outsider gets the uucpfriend login/password, masquerading is trivial.

But what does this have to do with the COMMANDS option, which only appears in MACHINE entries? It links 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 computer is logged in. In fact, it is an asynchronous process with no knowledge of what computer sent the execution request. Therefore, the real question is: How does your computer 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 the UUCP family of 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, or if the computer name does not appear in the Permissions file, the default list will be used.

The following example shows the relationship between the MACHINE and LOGNAME entries:

```
    MACHINE=eagle:owl:hawk REQUEST=yes \
    COMMANDS=rmail:/usr/lbin/rnews \
    READ=/  WRITE=/

    LOGNAME=uucpz VALIDATE=eagle:owl:hawk \
    REQUEST=yes SENDFILES=yes \
    READ=/  WRITE=/
```

The value in the COMMANDS option means that remote mail and /usr/lbin/rnews can be executed by remote users.

In the first entry, you must make the assumption that when you want to call one of the computers listed, you are really calling either eagle, owl, or hawk. Therefore, any file 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 will also be put in the privileged spool directory. You therefore have to validate that the computer has the privileged login uucpz.

MACHINE Entry for "Other" Systems

You may want to specify different option values for the computers your computer calls that are not mentioned in specific MACHINE entries. This may occur when there are many computers calling in and the command set changes from time to time. The name OTHER for the computer name is used for this entry as shown below:

```
MACHINE=OTHER \
COMMANDS=rmail:rnews:/usr/lbin/Photo:/usr/lbin/xp
```

All other options available for the MACHINE entry may also be set for the computers that are not mentioned in other MACHINE entries.

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 two entries

```
MACHINE=eagle:owl:hawk REQUEST=yes \
  READ=/  WRITE=/

LOGNAME=uucpz REQUEST=yes SENDFILES=yes \
  READ=/  WRITE=/
```

share the same `REQUEST`, `READ`, and `WRITE` options. These two entries can be merged as shown below:

```
MACHINE=eagle:owl:hawk REQUEST=yes \
LOGNAME=uucpz SENDFILES=yes \
  READ=/  WRITE=/
```

`Poll` File

The `Poll` file (`/etc/uucp/Poll`) contains information for polling remote computers. Each entry in the `Poll` file contains the name of a remote computer to call, followed by a TAB character (a space won't work), and finally the hours the computer should be called. The format of entries in the `Poll` file are:

*sys-name*TAB*hour* ...

For example the entry:

```
eagle   0       4       8       12      16      20
```

provides polling of the computer `eagle` every four hours.

The `uudemon.poll` script does not actually perform the poll. It merely sets up a polling work file (always named C.*file*) in the spool directory. `uudemon.hour` starts the scheduler, and the scheduler examines all work files in the spool directory.

`Devconfig` File

The `/etc/uucp/Devconfig` file is used when your computer communicates over a STARLAN network or some other network.

`Devconfig` entries define the STREAMS modules that are used for a particular device. Entries in the `Devconfig` file have the format:

```
service=x device=y push=z[:z . . .]
```

where *x* can be cu, uucico, or both separated by a colon; *y* is the name of a network and must match an entry in the Devices file; and *z* is replaced by the names of STREAMS modules in the order that they are to be pushed onto the Stream. Different modules and devices can be defined for cu and uucp services.

The following entries would most commonly be used in the file:

```
service=cu      device=STARLAN  push=ntty:tirdwr
service=uucico  device=STARLAN  push=ntty:tirdwr
```

This example pushes ntty, then tirdwr. The Devconfig file cannot be modified with the sysadm menu interface. If you want to change the contents of this file, you can use a text editor.

Sysfiles File

The /etc/uucp/Sysfiles file lets you assign different files to be used by uucp and cu as Systems, Devices, and Dialers files. Here are some cases where this optional file may be useful.

- You may want different Systems files so requests for login services can be made to different addresses from requests for uucp services.
- You may want different Dialers files to use different handshaking for cu and uucp.
- You may want to have multiple Systems, Dialers, and Devices files. The Systems file in particular may become large, making it more convenient to split it into several smaller files.

The format of the Sysfiles file is

```
service=w  systems=x[:x...] dialers=y[:y...] devices=z[:z...]
```

where *w* is replaced by uucico, cu, or both, separated by a colon; *x* is one or more files to be used as the Systems file, with each file name separated by a colon and read in the order presented; *y* is one or more files to be used as the Dialers file; and *z* is one or more files to be used as the Devices file. Each file is assumed to be relative to the /etc/uucp directory, unless a full path is given. A backslash (\) can be used to continue an entry on to the next line.

Here's an example of using a local Systems file in addition to the usual Systems file:

```
        service=uucico:cu        systems=Systems:Local_Systems
```

If this is in /etc/uucp/Sysfiles, then both uucico and cu will first look in /etc/uucp/Systems. If the system they're trying to call doesn't have an entry in that file, or if the entries in the file fail, then they'll look in /etc/uucp/Local_Systems.

When different Systems files are defined for uucico and cu services, your machine will store two different lists of Systems. You can print the uucico list using the uuname command or the cu list using the uuname -c command.

Limits File

The /etc/uucp/Limits file is used to control the maximum number of simultaneous uucicos, uuxqts, and uuscheds that are running in the uucp networking.

The format of the Limits file is

```
      service=x  max=y
```

where x can be uucico, uuxqt or uusched; and y is the limit that is permitted for that service.

The fields are order insensitive and lower case.

The following entries should most commonly be used in the file:

```
      service=uucico max=5
      service=uuxqt max=5
      service=uusched max=2
```

The example allows five uucicos, five uuxqts, and two uuscheds running on your machine. The Limits file cannot be modified with the System Administration Menus command sysadm. If you want to change the contents of this file, you must use one of the UNIX system text editors.

`Grades` File

The `Grades` file (`/etc/uucp/Grades`) contains the definitions for the job grades that may be used to queue jobs to a remote computer. It also contains the permissions for each job grade. Each entry in this file represents a definition of an administrator defined job grade that allows users to queue jobs.

Each entry in the `Grades` file has the following format:

User-job-grade System-job-grade Job-size Permit-type ID-list

Each entry in this file contains fields that are separated by white space. The last field in the entry is made up of sub-fields that are also white space separated. If a entry takes up more than one physical line, then a backslash (\) is used to continue the entry onto the following line. Comment lines begin with a pound sign (#) and occupy the entire line. Blank lines are always ignored. Here is a description of each field:

User-job-grade	This field contains an administrative defined user job grade name of up to 64 characters.
System-job-grade	This field contains a one character job grade to which *User-job-grade* will be mapped. The valid list of characters is A–Z,a–z, with A having the highest priority and z the lowest.
Job-size	This field specifies the maximum job size that can be entered in the queue. *Job-size* is measured in bytes and may be a list of the following:

nnnn	where *nnnn* is an integer that specifies the maximum job size for this job grade
*n*K	where *n* is a decimal number that represents the number of kilobytes and `K` is an abbreviation for kilobyte
*n*M	where *n* is a decimal number that represents the number of megabytes and `M` is an abbreviation for megabyte
`Any`	a keyword to specify that there is no maximum job size

Here are some examples:

`5000`	represents 5000 bytes.
`10K`	represents 10 kilobytes
`2M`	represents 2 megabytes.

Permit-type This field contains a keyword that denotes how to interpret the ID list. The following list contains the keywords and their meanings:

`User`	ID list contains the login names of users permitted to use this job grade.
`Non-user`	ID list contains the login names of users not permitted to use this job grade.
`Group`	ID list contains the group names whose members are permitted to use this group.
`Non-group`	ID list contains the group names whose members are not permitted to use this job grade.

Id-list This contains a list of login names or group names that are to be permitted or denied queuing to this job grade. The list of names are separated by white space and terminated by a newline character. The keyword `Any` is used to denote that anyone is permitted to queue to this job grade.

The user job grade may be bound to more than one system job grade. It is important to note that the `Grades` file will be searched sequentially for occurrences of a user job grade. Therefore, any multiple occurrences of a system job grade should be listed according to the restriction on the maximum job size.

While there is no maximum number for the user job grades, the maximum number of system job grades allowed is 52. The reason is that more than one *User-job-grade* can be mapped to a *System-job-grade,* but each *User-job-grade* job grade must be on a separate line in the `Grades` file. Here's an example:

```
mail        N     Any     User     Any
netnews     N     Any     User     Any
```

Given this configuration in a `Grades` file, these two *User-job-grade* grades will share the same *System-job-grade.* Since the permissions for a *Job-grade* are associated with a *User-job-grade* and not a *System-job-grade,* it is even possible for two *User-job-grades* to share the same *System-job-grades* and have two different sets of permissions for each one.

Default Grade

The binding of a default *User-job-grade* to a system job grade can be defined by the administrator. The administrator must use the keyword `default` as user job grade in the *User-job-grade* field of the `Grades` file and the system job grade that it is bound to. The Restrictions and id fields should be defined as `Any` so that any user and any size job can be queued to this grade. Here's an example:

```
default     a     Any     User     Any
```

If the default user job grade is not defined by the administrator, then the built-in default grade, `Z`, will be used. Because it is assumed that the restriction field is `Any`, multiple occurrences of the default grade is not checked.

`remote.unknown` File

There is one other networking file that affects the use of Basic Networking facilities, the `remote.unknown` file. This file is a binary program that executes when a machine not found in any of the `Systems` files starts a conversation. It will log the conversation attempt and drop the connection.

CAUTION If you change the permissions of the `remote.unknown` file so it cannot execute, your system will accept connections from any system.

Administrative Support Files

The Basic Networking administrative files are described below. These files are created in spool directories to lock devices, hold temporary data, or store information about remote transfers or executions.

`TM` (temporary data file)
: These data files are created by Basic Networking processes under the spool directory (that is, `/var/spool/uucp/`*X*) when a file is received from another computer. The directory *X* 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 or copied to the path name 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 *X* directory. These files should be automatically removed by `uucleanup`.

`LCK` (lock file)
: These lock files prevent duplicate conversations and transfers. The names have the form:

  `LCK.`*system.grade*

  where *system* is the name of the remote system and *grade* is the *System-Job-grade* being processed. The lock file contains the process ID of the process holding the lock. This process ID remains valid as long as the process is active.

`LK` (lock file)
: These lock files prevent duplicate use of calling devices. The names have the form:

  `LK.`*MAJ.maj.min*

  where *MAJ* is the major number of the device containing the directory entry. *maj* and *min* are the major and minor numbers, respectively, of the device itself. The lock file contains the process ID of the process holding the lock. This process ID remains valid as long as the process is active.

`C.` (work file) Work files are created in a spool directory when work (file transfers or remote command executions) has been 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:

- Type of request, `S` (send) or `R` (receive)
- Pathname of the file to be sent or received
- Pathname of the destination or user file name
- 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 may be 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 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 and *xxxx* is a four-digit job sequence number assigned by `uucp`. The four digit job sequence number may be followed by a sub-sequence number, *yyy*, which is used when there are several `D.` files created for a work (`C.`) file.

`P.` (Checkpoint file)
A checkpoint file is created when processing terminates abnormally. Unlike the `TM.`*pid.ddd* file, it is not removed. Therefore, when the transfer session is re-established, the length of the *checkpoint* file serves as an appropriate place to restart the transfer of the file, instead of restarting from the beginning. When checkpointing is specified for a file being transferred from another computer, the checkpoint file is used instead of a `TM` file. *checkpoint* files are created in the spool directory. Checkpointing occurs only between two systems that have the SVR4.0 BNU enhancements. The names of the checkpoint files have the following format:

`P.`*systmxxxyyy*

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 sub-sequence number, *yyy*, which is used when there are several `P.` files created for a work (`C.`) file. When the entire file is received, the `P.`*systmxxxyyy* file is moved to the path name specified in the `C.`*sysnxxxx* file (discussed above) that caused the transmission.

`X.` (Execute file) Execute files are created in the spool directory 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 number assigned by `uucp`. `Execute` files contain the following information:

- requester's login and computer name
- name of file(s) required for execution
- input filename to be used as the standard input to the command string
- computer and file name to receive standard output and stderr from the command execution
- command string
- option lines for return status requests

Logs

The BNU commands provide eight logs, some of which are optional. They are described below.

Command Log

The command log contains the commands issued by the user, the administrator, and the operator. It can help the system administrator in trouble-shooting. The full path name of the command log is `/var/spool/uucp/.Admin/command`. The format of each entry is as follows:

```
 user1  (5/16-19:00:31)  uucp test.c mach1!~/user2
|_____||________________||__________________________|
   a           b                      c
```

KEY:

a	user login name
b	date and time the command was issued
c	command line

System History Log

The system history log contains a record of each action that alters the state of the system and queue. The system history log can be generated by uucp, uucico, uux, or uuxqt programs and put into the uucp, uucico, uux, uuxqt subdirectories of the /var/spool/uucp/.Log directory. Below is a sample entry that uucico writes into /var/spool/uucp/.Log/uucico/mach1.

```
 uucp  mach1     (2/12-8:18:42,  2427,  0)  CONN FAILED
   a     b    c         d          e     f        g

 (NO DEVICES AVAILABLE)
            h
```

KEY:

a	user who submitted the job
b	name of the remote machine
c	job ID if a job is currently being processed; otherwise null, as in this line
d	date and time that the entry was written to the file
e	process ID of uucico in this example
f	file transfer sequence number within the current invocation of uucico
g	status message
h	status message elaboration

Error Log

The error log contains the error messages in the network. Error messages appear in the /var/spool/uucp/.Admin/errors file. When the errors occur, the program aborts. In most cases, this results from file-system problems. A typical entry is as follows:

```
ASSERT ERROR  (uucico)  pid: 1513  (5/15-9:03:45)  can't open
      a           b         c             d             e

system  3  [FILE:pkl.c,  LINE:356]
   f    g        h            i
```

KEY:

a	error type
b	program name
c	process ID
d	date and time that this entry was written to the file
e	error message part 1
f	error message part 2
g	error number
h	name of calling module
i	line of calling module where the error occurred

Transfer Log

The transfer log contains information pertaining to file transfer. For example, it shows the number of bytes transferred and how long the transfer took. After both `uucicos` (master and slave) agree on the protocol, the transfer information of each file is written into `/var/spool/uucp/.Admin/xferstats`. A typical entry is as follows:

```
ihx!  user  M  (5/20-6:10:10)  (C,  6559,  1)  [DKH]
  a     b   c        d           e     f    g     h

→ 1024/0.13 secs,  7877 bytes/sec  ""
         i               j          k
```

KEY:

a	name of remote system
b	user login
c	M=master, S=slave
d	date and time that entry was written to log
e	C=uucico, U=uucp, X=uux, Q=uuxqt
f	process ID of `uucico`, `uucp`, `uux`, or `uuxqt`
g	sequential number for each file transferred in a session
h	device name of media
i	direction and data bytes / clock time to transfer
j	transfer rate (bytes/sec)
k	`"PARTIAL FILE"` if the file was not completed because of transmission error; `""` if the file was transmitted completely (as in this record)

Report Statistics of File Transfer

By specifying the `-s`*file* option with `uucp`, you can have the statistics of your file transfer reported to *file*. For each file transfer request, *file* will contain three lines. For multiple file transfer requests, the size of *file* increases accordingly. A typical entry is:

First line:

$$\underbrace{\texttt{uucp job:}}_{a}\ \underbrace{\texttt{mach1N1131}}_{b}\ \underbrace{\texttt{(6/12-13:34:58)}}_{c}\ \underbrace{\texttt{(0:0:19)}}_{d}$$

KEY:

a	message header (always the same)
b	job ID assigned by `uucp`
c	date and time the file transfer completed
d	*hh:mm:ss* of elapsed time between time of creation of queue file and completion of the file transfer

Second Line:

$$\underbrace{\texttt{ihnp1!}}_{a}\ \underbrace{\texttt{/n15/user1/liblog}}_{b}\ \underbrace{\rightarrow}_{c}\ \underbrace{\texttt{mach1!}}_{d}\ \underbrace{\texttt{\textasciitilde/userid}}_{e}\ \underbrace{\texttt{(user1)}}_{f}$$

KEY:

a	sender node name
b	source file name
c	direction (always the same)
d	receiver node name
e	destination directory/file name
f	requester login ID

Third line (status):

$$\underbrace{\texttt{copy succeeded}}_{a}$$

KEY:

a	a message

Accounting Log

The accounting log contains information needed for network charging. Upon completing a successful job transaction, accounting information is written to file `/var/spool/uucp/.Admin/account`. If the job transaction is a file transfer, then accounting information is written to file `/var/spool/uucp/.Admin/account` on the requesting site. If the job transaction is a remote execution, then accounting information is written to file `/var/spool/uucp/.Admin/account` on the executing (target) site.

Accounting information is only collected if an account file exists and is writable by uucp; the file is not created automatically. As an administrator, you can create or remove the accounting log on your machine. A typical entry is as follows:

```
 5432  mach1N11152  4514  C  S  A  ihnp1  user1  (5/20-3:10:12)
|____||___________||____||_||_||_||_____||_____||______________|
  a         b         c    d  e  f    g      h           i

 mach1  user2  DKH  ""  xfer  ""
|_____||_____||___||__||____||__|
   j      k     l    m    n    o
```

KEY:

a	user ID
b	job ID assigned by uucp
c	size of job in bytes if the job transaction is a file transfer; time of job in seconds if the job transaction is a remote execution
d	C = job completed. P = partial job completed
e	service class of the job, namely: premium, standard, or economy. At present, only "standard" service class is supported.
f	job grade identification
g	originating system's name
h	originator's login name
i	date and time the job originated
j	destination system's name
k	destination user's login name
l	device name of media
m	ID of physical network (always "")
n	type of transaction: xfer = file transfer, rexe = remote execution.
o	The command if the job transaction is a remote execution.

Security Log

The security log contains the job transactions that attempt to violate system and user security measures. It is used to aid in detecting attacks on the systems. An attempted security violation is detected when the requester fails to pass the security checks specified in the `/etc/uucp/Permissions` file or tries to access a protected source or destination file. The occurrence is logged for further analysis in the `/var/spool/uucp/.Admin/security` file. Two different entries can appear in the security log.

`xfer`	file transfer
`rexe`	remote execution

Their formats are as follows:

$$\underbrace{\texttt{xfer}}_{a}\ \underbrace{\texttt{ihnp1}}_{b}\ \underbrace{\texttt{user1}}_{c}\ \underbrace{\texttt{mach1}}_{d}\ \underbrace{\texttt{user2}}_{e}\ \underbrace{\texttt{uucp.c}}_{f}\ \underbrace{\texttt{ihnp1}}_{g}\ \underbrace{\texttt{user1}}_{h}\ \underbrace{\texttt{uucp.c}}_{i}$$

$$\underbrace{\texttt{34567}}_{j}\ \underbrace{\texttt{(5/19-16:10)}}_{k}\ \underbrace{\texttt{(5/20-11:10:29)}}_{l}\ \underbrace{\texttt{(5/20-11:18:20)}}_{m}$$

KEY:

a	record type (always `xfer`)
b	requester node name
c	requester user login
d	destination node name
e	destination user login
f	destination file name
g	source node name
h	source file owner login
i	source file name
j	source file size in bytes
k	modification date and time of source file
l	date and time that transfer started
m	date and time that transfer completed

$$\underbrace{\texttt{rexe}}_{a}\ \underbrace{\texttt{ihnp1}}_{b}\ \underbrace{\texttt{user1}}_{c}\ \underbrace{\texttt{user2}}_{d}\ \underbrace{\texttt{(5/20-15:28:32)}}_{e}\ \underbrace{\texttt{(pwd)}}_{f}$$

KEY:

a	record type (always rexe)
b	client (requesting) node name
c	client (requesting) user login
d	server (destination) user login

e date and time that command was executed by server
f command name and options

Performance Log

The performance log contains statistics about the operation of `uucico`. `uucico` writes the log entries to `/var/spool/uucp/.Admin/perflog`. Statistics are only collected if `perflog` exists when `uucico` starts; the file is not created automatically. As an administrator, you can turn on or turn off performance logging at your machine. Two types of records will be written to the file; each is identified by a mnemonic type at the beginning of the record. The record types are:

NOTE Fields that are not applicable or unknown are marked by double quotes ("").

`conn`	contains statistics about the successful establishment of a connection
`xfer`	contains statistics about a file transfer.

Their formats are as follows:

```
 conn  860516175047  23517  mach1  M  ihnp1  DKH  d  ""
|____||____________||_____||_____||_||_____||___||_||__|
  a         b          c      d    e    f     g   h  i

 20.85  3.15  0.94
|_____||____||____
   j      k    l
```

KEY:

a record type (always conn)
b time stamp *YYMMDDhhmmss* for sorting
c uucico's process ID
d the name of the machine where the record was written

e	M = master, S = slave
f	name of remote system
g	device name of media
h	the protocol that was used for communications
i	physical network ID (always "")
j	real time to connect
k	user time to connect
l	system (kernel) time to connect

```
xfer  N   860516175051   23517  machl  M  ihnpl  DKH  d
 a    b        c           d      e    f    g     h   i

""  ihnplN2c76  118.00  121.00  1000  -dc  0.08  0.00
j       k          l       m      n     o    p     q

0.04  0.91  0.06  0.13  0.22  0.02  0.06  ""
 r     s     t     u     v     w     x    y
```

KEY:

a	record type (always `xfer`)
b	job grade ID
c	time stamp (*YYMMDDhhmmss*) for sorting
d	`uucico`'s process ID
e	the name of the machine where the record was written
f	M = master, S = slave
g	name of remote system
h	device name of media
i	the protocol that was used for communications
j	physical network ID (always "")
k	job ID if master, "" if slave
l	time in seconds that job was in queue if master, "" if slave
m	turn around time in seconds if master, "" if slave
n	size of the file that was actually transferred successfully or partially because of transmission error

o	command line options if master, "" if slave
p	real time to start up transfer
q	user time to start up transfer
r	system (kernel) time to start up transfer
s	real time to transfer file
t	user time to transfer file
u	system (kernel) time to transfer file
v	real time to terminate the transfer
w	user time to terminate the transfer
x	system (kernel) time to terminate the transfer
y	`"PARTIAL FILE"` if the file was not completed because of transmission error; "" if the file was transmitted completely (as in this record)

NOTE

Fields that are not applicable or unknown are marked by double quotes ("").

Start-up time includes the time for the master to search the queues for the next job, for the master and slave to exchange work vectors, and the time to open files.

Transfer time is the time it takes to transfer the data, close the file, and exchange confirmation messages.

Termination time is the time it takes to send mail notifications and write status files.

Turnaround time is the difference between the time that the job was queued and the time that the final notification was sent.

Foreign Log

The foreign log is a list of unknown systems that attempted to connect to the current machine. The List appears in the `/var/spool/uucp/.Admin/Foreign` file. The format produced by `remote.unknown` is as follows:

$$\underbrace{\texttt{Wed Jan 16 15:44:20 1987}}_{a}\ \underbrace{\texttt{udummy}}_{b}$$

KEY:

a	date and time that the entry was written to the file
b	the message logged by `remote.unknown`

8 Performance Management

An Overview of Performance Management

Performance in relation to a computer system is the way in which the system executes its tasks; its timeliness or responsiveness, including down time.

This chapter describes ways to monitor and enhance the performance of your UNIX system by telling you how to find and fix performance problems, by providing examples of how to improve performance, and by listing tools that monitor system performance. All these activities make up the task of performance management.

Performance management is a task, like all administrative tasks, that is a continual process. Usually it can be done routinely, on a regular schedule, but there are times when it will require your immediate attention.

In fact, performance management may need to be performed as soon as you set up your UNIX system. When you set up your system for the first time, it is automatically set to a basic configuration that is usually satisfactory for most applications. This default configuration controls the chief characteristics of your operating system, and this configuration may not take into consideration the traffic on your system nor the behavior of certain applications on your system. For this reason, you may need to reconfigure your system immediately in order to provide the service required by your users and their applications. For a full explanation of your system's default configuration, refer to your computer installation guide. Reconfiguration of your system will be described later in the chapter.

However, you may not know what your system's configuration should be at first. So, for argument's sake, we'll assume that you've accepted the default configuration.

Just as a car performs best when properly tuned, your system will perform best when it is properly tuned. For example, you may notice that the response time at the console is slow and something needs to be adjusted. Tuning is not just to correct performance problems. It is to maximize customer satisfaction.

At this point, you may want to utilize some of the tools which monitor performance to pinpoint the performance problem. These tools help you determine whether the problem is user-related or application-related, and are listed in the section "Monitoring System Performance."

Once you identify the problem, you will need to take some corrective action. Suggestions for these actions will be provided in the next section.

This is the extent of performance management. It is very possible that you may never need to do any special fine tuning to your system, and that your only experience with reconfiguration will be when you add new memory and peripherals. However, the rest of this chapter will be a handy reference when you need help recognizing performance problems and want advice on eliminating those problems.

Improving and Controlling System Performance

There are many items, both user-related and application-related, that can affect your system's performance. Possible problems and their solutions will be covered in this section. Keep in mind that any system tuning should be done during nonprime time.

Modifying the Tunable Configuration Parameters

Parameters exist which define your system's configuration; these parameters can be altered. This procedure is usually referred to as tuning the kernel, as you are adjusting the essential control structures at the heart of the system (the kernel). Use the `/usr/sbin/sysdef` shell command to see what the current parameter values are in the present configuration of your system. These parameters and their values are described in detail in your computer installation manual. How to tune these parameters is also explained in your computer installation manual. However, an example of this procedure appears at the end of this chapter.

Improving and Controlling File System Usage

Making files is easy under the UNIX operating system. Therefore, users tend to create numerous files using large amounts of space. The file systems containing the following directories should maintain, at the very least, the following start-of-day counts.

`/tmp`	2000 to 4000 512-byte blocks
`/usr`	500 to 1000 512-byte blocks
`/home`	3000 to 6000 512-byte blocks
`/var`	4000 to 8000 512-byte blocks

Other file systems should have 6 to 10 percent of their capacity available.

The default system configuration is set up so that the file blocks are allocated in an optimum manner. Refer to Chapter 5, "File System Administration," for more information on file system allocation.

Balancing File System Space

You can also control file system space by balancing the load between file systems. To do this, user directories often need to be moved. Users should be notified of moves well enough in advance so that they can program around the expected change.

In order to move directories and manipulate the file system tree, you must use the `find` and `cpio` commands. The following command sequence shows how to do this. This example moves directory trees userx and usery from file system `fs1` to `fs2` where, presumably, there is more space available.

```
cd /fs1
find userx usery -print -depth | cpio -pdm /fs2
rm -rf /fs1/userx /fs1/usery
```

Once this sequence is entered, verify that the copy was made. Then change the userx and usery default login directories with the usermod shell command. You must also notify userx and usery, preferably via `mail`, that they have been moved and that their pathname dependencies may need to be changed.

Whenever moving users in this way, make sure that users with common interests are in the same file system. Furthermore, move groups of users with a single cpio command, as shown in the example above, otherwise linked files will be unlinked and duplicated.

Selecting a File System Type

The default file system type is the S5 file system type with a logical block size of 2K (2048 bytes). For most applications this should be best. Depending on the average size of the file, however, you may want to change either the logical block size or even the file system type of the file system. There are three logical block sizes of S5 file systems: 512 byte, 1K (1024 bytes), and 2K (the default). The UFS file system has two logical block sizes: 4K (4096 bytes) and 8K (8192 bytes). Each has its advantages and disadvantages in terms of performance.

The UNIX kernel uses the logical block size when reading and writing files. If the logical block size of the file system is 2K, whenever I/O is done between a file and memory, 2K chunks of the file are read into or out of memory.

Large logical block size improves disk I/O performance by reducing seek time and also decreases CPU I/O overhead. On the other hand, if the logical block size is too large then disk space is wasted. The extra space is lost because even if only a portion of a block is needed the entire block is allocated. For example, if files are stored in 1K (1024 bytes) logical blocks, then a 24-byte file wastes 1000 bytes. If the same 24-byte file is stored on a file system with a 2K (2048 bytes) logical block size, then 2024 bytes are wasted. However, if most files on the file system are very large this waste is reduced considerably.

For a file system with mostly small files, small logical block sizes (512 byte and 1K) have the advantage of less wasted space on disk. However, CPU overhead may be increased for files larger than the block size.

The `sar -u` command, described later in the chapter, can help determine if large I/O transfers are slowing the system down.

Controlling Directory Size

Very large directories are very inefficient and can affect performance. Two directories in particular, `/var/mail` and `/var/spool/uucp`, tend to get very large and should be monitored periodically. If a directory becomes bigger than 10 logical blocks (forty, 512-byte blocks or 1280 entries for a 2K logical block size) or 2560 entries, whichever is smaller, then directory searches are likely causing performance problems. The `find` command, as shown below, can ferret out such problem directories.

```
find / -type d -size +40 -print
```

NOTE

`find` thinks in terms of 512-byte blocks.

Another important thing to remember is that removing files from directories *does not* make those directories any smaller. When a file is removed from a directory, the inode (file header) number is nulled out. This leaves an unused slot for that inode; over time the number of empty slots may become quite large. For example, if you have a directory with 100 files in it and you remove the first 99 files, the directory still contains the 99 empty slots, at 16 bytes per slot, preceding the active slot. In effect, unless a directory is reorganized on the disk, it will retain the largest size it has ever achieved.

If you have the 3B2 Cartridge Tape Utilities, you can run either the compress interface task or the /usr/sbin/dcopy shell command to compress the file system. This procedure copies the file system temporarily to the cartridge tape and then back to its original location.

If you don't have this utility package, you use the /usr/sbin/cmpress shell command. During reorganization, the system can be up but the file system being compressed must be unmounted. Root reorganization should be done once a week (requires a system reboot) and user file systems should be reorganized once a month in order to maintain maximum system performance.

An example of the dcopy command is:

```
/usr/sbin/dcopy fs1 fs2
```

where *fs1* and *fs2* can be the same name, however, the original *fs1* will be written over.

NOTE dcopy normally catches interrupts and quit signals and reports its progress. To kill dcopy, send it a quit signal followed by an interrupt (see dcopy(1M) for more information).

If you want to compress a single directory, you must perform a series of commands in order to perform this procedure. These commands are as follows:

```
mkdir /var/omail
mv /var/mail /var/omail
chmod 777 /var/omail
cd /var/omail
find . -print | cpio -plm ../mail
cd ..
rm -rf omail
```

You can also reduce directory size by locating inactive files, backing them up, and then deleting them. The find command can be used to locate inactive files. For example:

```
find / -mtime +90 -atime +90 -print > filename
```

where *filename* will contain the name of the files neither written to nor accessed within a specified time period. In this example, we used 90 days (+90). If these inactive files are causing problems, it is wise to first contact the user to see if these files can be deleted.

Controlling System Work Loads

Another step that can be performed to improve system performance is to check whether prime-time load can be reduced. You should control:

- less important jobs interfering with more important jobs
- scheduling of large jobs when the system is busy
- the efficiency of user-defined features, such as PATH variables

Controlling User PATH Variables

User PATH variables are the most difficult items to control. Regular mail should be sent to users on this subject informing them of how these items can cause system problems.

$PATH is a command line in the user's `.profile` file that is searched upon each command execution. Before outputting the `not found` error message, the system must search every directory in $PATH. These searches require both processor and disk time. If there is a disk or processor slowdown, changes here can help performance.

Some things that should checked for in user PATH variables are:

- $PATH is read left to right, so the most likely places to find the command should be first in the path. Make sure that a directory is not searched more than once for a command.
- Users may prefer to have the current directory listed first in the path (`:/usr/bin`).
- In general, $PATH should have the least number of required entries.
- Searches of large directories should be avoided if possible. Put any large directories at the end of $PATH.

- Directories that are actually a symbolic link to another directory should not appear in `$PATH`, for example, `/usr/bin` should not be in `$PATH`, but rather `/usr/bin` should.

Controlling Runaway Processes

The ps command is used to obtain information about active processes. This command gives a "snapshot" of what is going on, which is useful when you are trying to identify what processes are loading the system. The entries you should be interested in are `TIME` (minutes and seconds of CPU time used by processes) and `STIME` (time when the process first started).

When you spot a runaway process (one that uses progressively more system resources over a period of time while you are monitoring it), you should check with the owner. It is possible that such a process should be terminated immediately via the `kill -9` shell command. When you have a real runaway, it continues to eat up system resources until everything grinds to a halt.

When you spot processes that take a very long time to execute you should consider using the `cron` or `at` command to execute the job during off-hours.

Monitoring System Performance

The need to improve and control system usage may not be evident unless you monitor your system regularly. For example, it may not be obvious that a system is degraded. Just as a driver might not notice the difference between 48 and 50 miles per hour without the aid of a speedometer, you might not notice a 4 percent degradation without performance monitoring. This section will show you many ways to monitor your system's performance.

df and du Usage Reports

You can monitor your file system usage by executing the df command regularly during the day. The df command prints out the number of free file blocks and inodes.

The du command can be executed daily after hours. The du command summarizes file system usage with a total for each directory being printed out. Note that if there are links between files in different directories, where the directories are on separate branches of the file system hierarchy, du will count the excess files more than once.

The output from these commands can be kept for later comparison. In this way, directories which rapidly increase their space usage can be spotted. However, these reports may not provide the amount of detail that you need to pinpoint exact performance problems. The following section describes monitoring tools that do provide a high amount of detail.

System Performance Analysis Utilities (SPAU) Tools

The tools within SPAU provide you with commands for collecting and examining system usage data. These reports can be used to analyze the current performance of the computer and determine load-balancing and system-tuning strategies that will improve performance.

The sections, "Kernel Profiling" and "System Activity Reporting," "System Activity Reporting," describe each SPAU command, its purpose, the use of its options, and examples of command usage. The examples provided are from a 3B2 system with 4 megabytes of main memory and a 72-megabyte hard disk. The values you receive may be different. However, the output you receive should help you determine traffic problems and whether they are user- or application-related.

Installing SPAU

Before you install the appropriate SPAU package, you must remove any release of performance management utilities that may have been previously installed. Refer to your computer installation manual for the procedure to install and remove software.

Check the amount of free space in the `root` and `/usr` directories before installation. This package requires 1 block in `root` and 618 blocks in `/usr`.

Summary of SPAU commands

SPAU contains ten commands and two shell scripts that are listed below:

`prfdc`	Performs the data collection function of the profiler by copying the current value of all the text address counters to a file where the data can be analyzed.
`prfld`	Used to initialize the recording mechanism in the system.
`prfpr`	Formats the data collected by `prfdc` or `prfsnap`.
`prfsnap`	Collects data (like `prfdc`) at the time of invocation only.
`prfstat`	Used to enable, disable, or check the status of the sampling mechanism.
sadc	Used to sample and save the system activity data.
sadp	Reports disk access location and seek distance in tabular form.
sag	Graphically displays the information collected by `sar`.
`sar`	Calls sadc or uses files created by sadc to sample cumulative activity counters internal to the UNIX system and provides reports on various system-wide activities. Results are saved in binary format.
`sa1`	Shell script used to collect and store data in binary file `/var/adm/sa/sa`*dd* where *dd* is the current day.
sa2	Shell script that writes a daily report in file `/var/adm/sa/sar`*dd* where *dd* is the current day.

`timex`	When `timex` is used to execute another command, the elapsed time, user time, and system time spent in execution of the command are reported in seconds.

To use any of the commands beginning with `prf`, you must be logged in as `root`. The same is true for both `sa1` and `sa2`.

The next two sections describe the SPAU tools in detail.

Kernel Profiling

Kernel profiling is a mechanism that allows you to determine where the operating system is spending its time during operation. It consists of commands that control the profiling process and generate reports (see `profiler(1M)` for a description of these commands). The system profiler samples the program counter on every clock interrupt and increments the counter corresponding to the function shown by that value of the program counter.

The system profiler initializes the sampling mechanism. It will then generate a table containing the starting address of each system subroutine as extracted from the UNIX system kernel. To operate the system profiler, you must perform the following steps:

1. Use the `prfld` command to initialize or load the profiler.
2. Use the `prfstat` command to turn on the sampling mechanism.
3. Use the `prfdc` or `prfsnap` command to collect and enter the data into a file.
4. Use the `prfpr` command to print the contents of the data, collected by either `prfdc` or `prfsnap`.
5. Use the `prfstat` command to turn off the sampling mechanism.

The system profiler must be loaded and turned on after every boot. If you want the profiler to begin automatically when you boot the system to multi-user mode, you can add the following lines to the `/sbin/init.d/perf` file:

```
/usr/sbin/prfld
/usr/sbin/prfstat on
```

The `prf` shell script will be executed during system initialization and the following messages will be displayed:

```
profiling enabled

xxx kernel text addresses
```

where *xxx* states how many kernel text addresses are in the current UNIX system kernel.

The following sections describe kernel profiling commands in detail.

Loading the System Profiler

The `prfld` command initializes, or loads, the system profiler mechanism. The command has the following format:

```
        /usr/sbin/prfld [namelist]
```

This command generates a table, in memory, containing the starting address of each subroutine as extracted from *namelist*. The default of *namelist* is `/stand/unix`. If *namelist* is not indicated, the starting address of each subroutine is recorded. If the number of kernel text addresses is greater than `PRFMAX` defined in `/etc/master.d/prf`, then `PRFMAX` should be increased and the system rebuilt (see the section, "Reconfiguring the System Through a Reboot").

Enabling/Disabling the Sampling Mechanism

The `prfstat` command enables or disables the sampling mechanism of the system profiler initialized by `prfld`. The `prfstat` command has the following format:

```
        /usr/sbin/prfstat [on | off]
```

Profiler overhead is less than 3 percent as calculated for 2000 text addresses. If neither of the optional parameters is entered, the status of the profiler is displayed. If the on parameter is supplied, the sampling mechanism is turned on. The opposite happens if `off` is indicated.

Collecting Profiling Data

The `prfdc` command performs the data collection function of the profiler by copying the current value of all the text address counters to a file where the data can be analyzed. The `prfdc` command has the following format:

```
        /usr/sbin/prfdc file [period[off_hour]]
```

This command stores the contents of the counters in a *file* every *period* minutes and turns off at *off_hour*. Valid values for *off_hour* are 0 through 24.

For example, the following copies the current value of all the text address counters into a file called `temp` every five minutes and turns off at 4:00 P.M.:

```
/usr/sbin/prfdc temp 5 16
```

The `prfsnap` command also performs data collection, but takes a snapshot of the system at the time it is called. The format of this command is as follows:

```
/usr/sbin/prfsnap [file]
```

where the command appends the counter values to *file*.

Formatting the Collected Data

The `prfpr` command formats the contents of *file* (data that was collected by `prfdc` or `prfsnap`). The `prfpr` command has the following format:

```
/usr/sbin/prfpr file [cutoff [namelist] ]
```

Each text address is converted to a system function name and the percentage of time used by that function is printed if the the activity percentage is greater than the *cutoff* number that you specify. The range of *cutoff* is 0 percent to 99 percent where 0 prints all contents. The default *cutoff* is 1 percent. The default *namelist* is `/stand/unix`.

The following screen display illustrates the output of the `prfpr` command.

```
# /usr/sbin/prfpr temp 0

07/20/89 18:02
07/20/89 18:03

_waitloc 91.0
fsflush 0.3
bdwrite 0.1
brelse 0.1
getblk 0.1
dma_access 0.1
wakeprocs 0.2
systrap 0.1
mon_enter 0.1
ts_setdq 0.1
iupdat 0.3
idstrategy 0.2
idsetup 0.4
idint 0.2
idldcmd 0.1
user     0.7
```

These are the function calls in the kernel. For detailed information on function calls, refer to your computer installation manual, source code, or experienced user.

System Activity Reporting

Another SPAU tool is system activity reporting. As various system actions occur, counters in the operating system are incremented to keep track of these activities. System activities that are tracked are:

- central processing unit (CPU) utilization
- buffer usage
- disk and tape input/output activity
- terminal device activity
- system call activity
- switching
- file access
- queue activity
- kernel tables
- interprocess communication
- paging
- free memory and swap space
- Kernel Memory Allocation (KMA)
- Remote File Sharing (RFS)

System activity data can be accessed on a special request basis using the `sar` command or it can be saved automatically on a routine basis using the `sadc` command and the shell scripts `sa1` and `sa2` (see `sar`(1M)). Generally, the demand system activity reports are used to pinpoint specific performance problems, and the automatic reports are generated as a measure to monitor system performance.

The following sections describe both methods of activity reporting in detail.

Automatically Collecting System Activity Data

The sadc command can automatically sample system data. The format of this command is as follows:

```
/usr/lib/sa/sadc [t n] [ofile]
```

The command samples *n* times with an interval of *t* seconds (*t* should be greater than 5 seconds) between samples. It then writes, in binary format, to the file *ofile*, or to standard output. If *t* and *n* are omitted, a default interval is used.

When the performance package is installed, a number of files should be automatically created and/or appended that will cause system activity commands to be run automatically.

The file /sbin/init.d/perf, which is linked to /sbin/rc2.d/S21perf, causes the sadc command to be invoked to mark usage from when the counters are reset to zero. The output of sadc is put in the file sa*dd* which acts as the daily system activity record. The command entry in the /sbin/init.d/perf file that does this is as follows:

```
su sys -c "/usr/lib/sa/sadc /var/adm/sa/sa`date +%d`"
```

Once the performance package is installed, the cron file /var/spool/cron/crontabs/sys contains commands to cause the automatic collection of system activity data. The commands in the cron file are sa1 and sa2. The shell script sa1 has the following format:

```
/usr/lib/sa/sa1 [t n]
```

The arguments *t* and *n* cause records to be written *n* times at an interval of *t* seconds. If these arguments are omitted, the records are written only one time. The records are written to the binary file /var/adm/sa/sa*dd*, where *dd* is the current date. The sa1 command is performed automatically by cron using the following two entries found in /var/spool/cron/crontab/sys:

```
0 * * * 0-6 /usr/lib/sa/sa1
20,40 8-17 * * 1-5 /usr/lib/sa/sa1
```

The first causes a record to be written to /var/adm/sa/sa*dd* on the hour, every hour, seven days a week. The second entry causes a record to be written to /var/adm/sa/sa*dd* 20 minutes and 40 minutes after each hour from 8:00 a.m. to 5:00 p.m., Monday through Friday, typically considered to be peak working hours. Thus, these two crontab entries cause a record to be written to /var/adm/sa/sa*dd* every 20 minutes from 8:00 a.m. to 5:00 p.m., Monday through Friday, and every hour on the hour otherwise. These defaults can easily be changed to meet your daily needs.

The shell script sa2 has the following format:

```
    /usr/lib/sa/sa2 [-abcdgkmpqruvwxyADSC] [-s time] [-e time] [-i sec]
```

The sa2 command invokes the sar command with the arguments given and writes the ASCII output to the file /var/adm/sa/sar*dd* where *dd* is the current date. The report starts at -s *time*, ends at -e *time*, and is taken as close to -i *sec* intervals as possible. See the sar command, later in the chapter, for an explanation of the remaining options.

When installed, the performance package includes the following entry in the /var/spool/cron/crontabs/sys file:

```
    5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200 -A
```

This causes a sar -A report to be generated from /var/adm/sa/sa*dd*. The report covers twenty-minute intervals in the time period from 8:00 a.m. to 6:01 p.m., Monday through Friday. Note that since /var/adm/sa/sa*dd* does not have data for 5:20 and 5:40 if the above sa1 cron entries are used, that the sar report will not have data for those times either.

Collecting System Activity Data on Demand

The sar command can either be used to gather system activity data itself or to extract what has been collected in the daily activity files created by sa1 and sa2.

The sar command has the following formats:

```
    sar [-abcdgkmpqruvwxyADSC] [-o file] t [n]
    sar [-abcdgkmpqruvwxyADSC] [-s time] [-e time] \
        [-i sec] [-f file]
```

In the first format, sar samples cumulative activity counters in the operating system at intervals specified by *n* for a time (in seconds) specified by *t* (*t* should be 5 seconds or greater). The default value of *n* is 1. If the -o option is specified, samples are saved in *file* in binary format.

In the second format, with no sampling interval specified, `sar` extracts data from a previously recorded *file*, either the one specified by the `-f` option or, by default, the standard daily activity file, `/var/adm/sa/sar`*dd*. The `-s` and `-e` options define the starting and ending times for the report. Starting and ending times are of the form *hh[:mm[:ss]]*. The `-i` option specifies, in seconds, the intervals to select records. If the `-i` option is not included, all intervals found in the daily activity file are reported.

The following summarizes `sar` options and their results:

`-a`	checks file access operations
`-b`	checks buffer activity
`-c`	checks system calls
`-d`	checks disk activity
`-g`	checks page-out and memory freeing
`-k`	checks kernel memory allocation
`-m`	checks interprocess communication
`-p`	checks page-in and fault activity
`-q`	checks queue activity
`-r`	checks unused memory
`-u`	checks CPU utilization
`-v`	checks system table status
`-w`	checks swapping and switching volume
`-y`	checks terminal activity
`-A`	reports overall system performance; same as entering all options

The options `-x`, `-D`, `-S`, and `-C` are remote file sharing options, and are described in Chapter 7, "Network Services."

Checking File Access with `sar -a`

The `sar -a` option reports on the use of file access operations. The UNIX operating system routines reported are as follows:

`iget/s`	Number of S5 and UFS files located by inode entry per second.
`namei/s`	Number of file system path searches per second. If `namei` does not find a directory name in the directory name logic cache, it will call `iget` to get the directory. Hence, most `igets` are the result of directory name logic cache misses.
`dirbk/s`	Number of S5 directory block reads issued per second.

The following is an example of `sar -a` output. It illustrates a one-minute sampling interval.

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12  iget/s namei/s dirbk/s
14:29:12       0       2       1
14:30:12       0       4       1
14:31:12       0       3       1

Average        0       3       1
```

The larger the values reported, the more time the UNIX kernel is spending to access user files. The amount of time reflects how heavily programs and applications are using the file system(s). The `-a` option is helpful for understanding how disk- dependent an application system is; it is not used for any specific tuning step.

Checking Buffer Activity with `sar -b`

The `-b` option reports on the following buffer activities.

`bread/s`	Average number of physical block (512 bytes each) reads into the system from the disk per second.

`lread/s`	Average number of logical reads from system buffers per second.
`%rcache`	Fraction of logical reads found in the system buffers (100% minus the ratio of `breads` to `lreads`).
`bwrit/s`	Average number of physical writes from the system buffers to disk per second.
`lwrit/s`	Average number of logical writes to system buffers per second.
`%wcache`	Fraction of logical writes found in the system buffers (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 most important entries are the cache hit ratios `%rcache` and `%wcache`, which measure the effectiveness of system buffering. If `%rcache` falls below 90, or if `%wcache` falls below 65, it may be possible to improve performance by increasing the buffer space by adjusting the tunable `BUFHWM` in `/etc/master.d/kernel`.

The following is an example of `sar -b` output:

```
14:28:12 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
14:29:12       0      14     100       6      17      67       0       0
14:30:12       0      12      99       6      16      65       0       0
14:31:12       0      12     100       6      16      65       0       0

Average        0      12     100       6      16      66       0       0
```

This example shows that the buffers are not causing any slowdowns, because all the data are within acceptable limits.

A parallel option is available for systems that have Remote File Sharing installed. See the description of `sar -Db` in the "Network Services" chapter.

Checking System Calls with `sar -c`

The `-c` option reports on system calls in the following categories:

`scall/s`	All types of system calls per second (generally about 30 per second on a busy four- to six-user system).
`sread/s`	`read` system calls per second.
`swrit/s`	`write` system calls per second.
`fork/s`	`fork` system calls per second, about 0.5 per second on a four- to six-user system. This number will increase if shell scripts are running.
`exec/s`	`exec` system calls per second. (If `(exec/s)` / `(fork/s)` is greater than 3, look for inefficient `PATH` variables.)
`rchar/s`	Characters (bytes) transferred by `read` system calls per second.
`wchar/s`	Characters (bytes) transferred by `write` system calls per second.

Typically, `reads` plus `writes` account for about half of the total system calls, although the percentage varies greatly with the activities that are being performed by the system.

This is an example of `sar -c` output:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
14:29:12      17       2       2    0.28    0.28    2527    1542
14:30:12      25       2       1    0.50    0.47    1624     295
14:31:12      21       2       2    0.35    0.35    1812     703

Average       21       2       2    0.38    0.37    1987     847
```

A parallel option is available for systems that have Remote File Sharing installed. See the description of `sar -Dc` in the "Network Services" chapter.

Checking Disk Activity with `sar -d`

The `sar -d` option reports the activities of disk devices.

`device`	Name of the disk device(s) monitored.
`%busy`	Percentage of time the device spent servicing a transfer request.
`avque`	The average number of requests outstanding during the monitored period (measured only when the queue was occupied).
`r+w/s`	Number of read and write transfers to the device per second.
`blks/s`	Number of 512 byte blocks transferred to the device per second.
`avwait`	Average time in milliseconds that transfer requests wait idly in the queue (measured only when the queue is occupied).
`avserv`	Average time in milliseconds for a transfer request to be completed by the device (for disks this includes seek, rotational latency, and data transfer times).

The following two examples illustrate the `sar -d` output. The first example is from a computer with a non-SCSI (Small Computer System Interface) integral disk, that is, a disk that does not use an SCSI interface. This example illustrates data being transferred from a hard disk (`hdsk-0`) to the floppy disk (`fdsk-0`).

The second example is from a computer with SCSI integral disks, that is, disks that use an SCSI interface. The example illustrates data being transferred from one SCSI hard disk (`sd00-0`) to another SCSI integral disk (`sd00-1`):

```
unix unix 4.0 2 3B2    8/11/89
13:46:28  device %busy avque r+w/s blks/s  avwait  avserv
13:46:58  hdsk-0     6   1.6     3      5    13.8    23.7
          fdsk-0    93   2.1     2      4   467.8   444.0
13:47:28  hdsk-0    13   1.3     4      8    10.8    32.3
          fdsk-0   100   3.1     2      5   857.4   404.1
13:47:58  hdsk-0    17    .7     2     41      .6    48.1
          fdsk-0   100   4.4     2      6  1451.9   406.5
Average   hdsk-0    12   1.2     3     18     8.4    34.7
          fdsk-0    98   3.2     2      5   925.7   418.2
```

SCSI integral disk:

```
unix unix 4.0 2 3B2    8/11/89
14:16:24  device %busy avque r+w/s blks/s  avwait  avserv
14:16:52 sd00-0      2   1.0     1      3     0.0    17.9
         sd00-1      6   1.1     3      5     2.0    23.9
14:17:21 sd00-0      2   1.0     1      2     0.0    19.6
         sd00-1      6   1.1     3      5     0.2    24.3
14:17:48 sd00-0      3   1.0     1      3     0.3    18.3
         sd00-1      7   1.1     3      5     1.3    25.4
14:18:15 sd00-0      3   1.0     1      3     0.0    17.2
         sd00-1      5   1.0     2      5     0.0    21.6
Average  sd00-0      2   1.0     1      3     0.1    18.2
         sd00-1      6   1.0     3      5     0.9    23.8
```

Note that queue lengths and wait times are measured when there is something in the queue. If `%busy` is small, large queues and service times probably represent the periodic `sync` efforts by the system to ensure that altered blocks are written to the disk in a timely fashion.

Checking Page-Out and Memory Freeing Activity with `sar -g`

The `sar -g` option reports page-out and memory freeing activities as follows:

`pgout/s`	The number of times per second file that systems receive page-out requests.

`ppgout/s`	The number of pages that are paged-out per second. (A single page-out request may involve paging-out multiple pages.)
`pgfree/s`	The number of pages per second that are placed on the freelist by the page-stealing daemon. If this value is greater than 5, it may be an indication that more memory is needed. (This is the same as `rclm/s` previously reported by option `-p`.)
`pgscan/s`	The number of pages per second scanned by the page-stealing daemon. If this value is greater than 5, the page-out daemon is spending a lot of time checking for free memory. This implies that more memory may be needed.
`%s5ipf`	The percentage of S5 inodes taken off the freelist by `iget` which had reusable pages associated with them. These pages are flushed and cannot be reclaimed by processes. Thus, this is the percentage of `igets` with page flushes. If this value is greater than 10 percent, then the freelist of inodes is considered to be page-bound and the number of S5 inodes should be increased.

The following is an example of `sar -g` output:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12  pgout/s ppgout/s pgfree/s pgscan/s %s5ipf
14:29:12     0.00     0.00     0.35     8.18   0.00
14:30:12     0.00     0.00     0.00     0.00   0.00
14:31:12     0.00     0.00     0.00     0.00   0.00

Average      0.00     0.00     0.12     2.72   0.00
```

`sar -g` is a good indicator of whether more memory may be needed. The number of cycles used by the page-stealing daemon can be found using `ps -elf`. If it has used many cycles then coupled with high `pgfree/s` and `pgscan/s` values it is a good indicator of a memory shortage. `sar -p`, `sar -u`, `sar -r`, and `sar -w` are also good memory shortage indicators.

`sar -g` also shows whether inodes are being recycled too quickly, causing a loss of reusable pages.

Checking Kernel Memory Allocation Activity with `sar -k`

The `-k` option reports on the following activities of the Kernel Memory Allocator (KMA).

`sml_mem`	The amount of memory in bytes the KMA has available in the small memory request pool (a small request is less than 256 bytes).
`alloc`	The amount of memory in bytes the KMA has allocated from its small memory request pool to small memory requests.
`fail`	The number of requests for small amounts of memory that failed.
`lg_mem`	The amount of memory in bytes the KMA has available in the large memory request pool (a large request is from 512 bytes to 4K bytes).
`alloc`	The amount of memory in bytes the KMA has allocated from its large memory request pool to large memory requests.
`fail`	The number of requests for large amounts of memory that failed.
`ovsz_alloc`	The amount of memory allocated for oversized requests (those greater than 4K). These requests are satisfied by the page allocator; thus, there is no pool.
`fail`	The number of requests for oversized amounts of memory that failed.

The KMA allows a kernel subsystem to allocate and free memory as needed. Rather than statically allocating the maximum amount of memory it is expected to require under peak load, the KMA divides requests for memory into three categories: small (less than 256 bytes), large (512 - 4K bytes), and oversized (greater than 4K bytes). It keeps two pools of memory to satisfy small and large requests. The oversized requests are satisfied by allocating memory from the system page allocator.

If your system is being used to write drivers or STREAMS that use KMA resources then `sar -k` will likely prove useful. Otherwise, you will probably not need the information it provides. Any driver or module that uses KMA resources but does not specifically return the resources before it exits can create a memory leak. A memory leak will cause the amount of memory allocated by KMA to increase over time. Thus, if the `alloc` fields of `sar -k` increase steadily over time, then there may be a memory leak. Another indication of a memory leak is failed requests. If this occurs then it is likely that a memory leak has caused KMA to be unable to reserve and allocate memory.

If it appears that a memory leak has occurred, you should check any locally written drivers or STREAMS that may have requested memory from KMA and not returned it.

The following is an example of `sar -k` output:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 sml_mem   alloc  fail  lg_mem   alloc  fail  ovsz_alloc  fail
14:29:12   95232   73472     0  311296  198656     0      180224     0
14:30:12   95232   75120     0  311296  198656     0      180224     0
14:31:12   95232   73600     0  311296  197632     0      180224     0

Average    95232   74064     0  311296  198314     0      180224     0
```

Checking Interprocess Communication with `sar -m`

The `sar -m` option reports interprocess communication activities. Message and semaphore calls are reported as follows:

`msg/s`	Number of message operations (sends and receives) per second.
`sema/s`	Number of semaphore operations per second.

An example of `sar -m` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12   msg/s  sema/s
14:29:12    0.00    0.00
14:30:12    0.00    0.00
14:31:12    0.00    0.00

Average     0.00    0.00
```

These figures will usually be zero (0.00) unless you are running applications that use messages or semaphores.

Checking Page-In Activity with `sar -p`

The `sar -p` option reports paging-in activity which includes protection and validity faults. (Note: This option has changed significantly from past releases due to the adoption of Virtual Memory.)

`atch/s`	The number of page faults per second that are satisfied by reclaiming a page currently in memory (attaches per second). Instances of this include reclaiming an invalid page from the free list and sharing a page of text currently being used by another process (for example, two or more processes accessing the code for data.)
`pgin/s`	The number of times per second file systems receive page-in requests. (This encompasses and replaces the old `sar -p` report of `pgfil/s`, which previously reported the number of validity faults per second satisfied by a page-in from the file system.)
`ppgin/s`	This new field reports the number of pages paged in per second. (A single page-in request, such as a softlock request as described below, or a large block size, may involve paging-in multiple pages.)

`pflt/s`	The number of page faults from protection errors per second. Instances of protection faults are illegal access to a page and "copy-on-writes." Generally, this number consists primarily of "copy-on-writes." (This field is carried over from the old -p option.)
`vflt/s`	The number of address translation page faults per second. These are known as validity faults and occur when a valid page is not present in memory. (This field is carried over from the old -p option.)
`slock/s`	This new field reports the number of faults per second caused by software lock requests requiring physical I/O. An example of the occurrence of a softlock request is the transfer of data from a disk to memory. To ensure that the page which is to receive the data is not claimed and used by another process, it is locked by the system hardware.

The following is an example of `sar -p` output:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
14:29:12    1.17   12.87   12.87    5.67   11.28    1.15
14:30:12    1.67    7.08    7.08    9.12    6.33    0.67
14:31:12    1.37   12.48   12.48    6.83   10.78    1.03

Average     1.40   10.81   10.81    7.21    9.46    0.95
```

If `vflt/s` becomes much higher than 15, then `sar -g` should be looked at to determine if there is a memory shortage or if the S5 inode freelist is page bound. (See `sar -g` for more details). In addition, `sar -u`, `sar -w`, and `sar -r` can help verify whether memory is a bottleneck.

Checking Queue Activity with `sar -q`

The `sar -q` option reports the average queue length while the queue is occupied and the percentage of time that the queue is occupied.

`runq-sz`	The number of processes waiting, in memory, to run. 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, the better.
`swpq-sz,`	Values for these headings are no longer reported due to the removal of swap queues.

An example of `sar -q` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 runq-sz %runocc swpq-sz %swpocc
14:29:12     1.2      53
14:30:12     1.3      38
14:31:12     1.1      37

Average      1.2      43
```

If `%runocc` is greater than 90 percent 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 `sar -p` shows a large number of validity faults and `sar -g` shows high page-out activity, then more memory may be required.

Checking Unused Memory with `sar -r`

The `-r` option records the number of memory pages and swap file disk blocks that are currently unused.

`freemem`	Average number of 2K pages of memory available to user processes over the intervals sampled by the command.

`freeswap` Number of 512-byte disk blocks available for page swapping.

An example of `sar -r` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 freemem freeswp
14:29:12     268    3034
14:30:12     351    3009
14:31:12     297    3033

Average      306    3025
```

Checking CPU Utilization with `sar -u`

The CPU utilization is listed by `sar -u`. 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 either waiting for input/output completion or "sitting still" with no work to do. `sar -u` lists the percentage of time that the processor is in system mode (`%sys`), in user mode (`%user`), waiting for input/output completion (`%wio`), and idle (`%idle`).

In typical timesharing use, `%sys` and `%usr` are about the same value. In special applications, either of these may be larger than the other without anything being abnormal. A high `%wio` generally means a disk slowdown has occurred. A high `%idle`, with degraded response time, may mean memory constraints are present; time spent waiting for memory is attributed to `%idle`.

The following is an example of `sar -u` output:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12    %usr    %sys    %wio   %idle
14:29:12      22      27      18      32
14:30:12       6      24      13      57
14:31:12       8      28      19      45

Average       12      27      17      45
```

If your 3B2 Computer is equipped with a co-processor, `sar -u` produces additional output for it, including an extra column showing the number of system calls per second being executed on the co-processor. The %wio column is empty for the co-processor because the co-processor does not perform I/O. The following `sar` data were collected on a co-processor system at intervals of 20 seconds.

```
sf600 sf600 4.0 3 3B2    08/12/89

10:02:07    %usr    %sys    %wio   %idle
10:02:27      82      18       0       0
10:02:47      39      35      16      10
10:03:07       7      28      16      50
10:03:27       1      16       0      83

Average       32      24       8      36

10:02:07 %co-usr %co-sys        %co-idle scall/s
10:02:27      98       0               2       0
10:02:47      65       0              35       0
10:03:07      11       0              89       1
10:03:27       0       0             100       0

Average       44       0              56       0
```

A parallel option is available for systems on which Remote File Sharing is installed. See the description of `sar -Du` in Chapter 7, "Network Services."

Checking System Table Status with `sar -v`

The `-v` option reports the status of the process, inode, file, and shared memory record table. From this report you know when the system tables need to be modified.

`proc-sz`	Number of process table entries currently being used/allocated in the kernel.
`inod-sz`	Number of inode table entries currently being used/allocated in the kernel.
`file-sz`	Number of file table entries currently being used in the kernel. The `sz` is given as 0 since space is allocated dynamically for the file table.
`ov`	Number of times a table has overflowed (reported for the three tables listed above).
`lock-sz`	Number of shared memory record table entries currently being used/allocated in the kernel. The `sz` is given as 0 because space is allocated dynamically for the shared memory record table.

An example of `sar -v` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 proc-sz ov inod-sz ov file-sz ov lock-sz
14:29:12  28/200  0 297/300  0  63/  0  0   6/  0
14:30:12  30/200  0 297/300  0  65/  0  0   6/  0
14:31:12  28/200  0 296/300  0  63/  0  0   6/  0
```

This example shows that all tables are large enough to have no overflows. If the values in these tables never exceed those shown here, you could reduce the sizes of the tables as a way of saving space in main memory.

Checking Swapping and Switching Volume with `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 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, you may need to increase memory or decrease the amount of buffer space.
`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 sfxbs 4.0 2 3B2    08/22/89

14:28:12 swpin/s pswin/s swpot/s pswot/s pswch/s
14:29:12    0.00     0.0    0.00     0.0      22
14:30:12    0.00     0.0    0.00     0.0      12
14:31:12    0.00     0.0    0.00     0.0      18

Average     0.00     0.0    0.00     0.0      18
```

This example shows that because no swapping is occurring, there is sufficient memory for the currently active users.

Checking Terminal Activity with `sar -y`

The –y option monitors terminal device activities. If you have a lot of terminal I/O, you can use this report to determine if there are any bad lines. The activities recorded are defined as follows:

`rawch/s`	input characters (raw queue) per second
`canch/s`	input characters processed by canon (canonical queue) per second
`outch/s`	output characters (output queue) per second
`rcvin/s`	receiver hardware interrupts per second
`xmtin/s`	transmitter hardware interrupts per second
`mdmin/s`	modem interrupts per second

The number of modem interrupts per second (`mdmin/s`) should be close to 0, and the receive and transmit interrupts per second (`xmtin/s` and `rcvin/s`) should be less than or equal to the number of incoming or outgoing characters, respectively. If this is not the case, check for bad lines.

An example of `sar -y` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
14:29:12       0       1     157       1       3       0
14:30:12       0       2      34       2       2       0
14:31:12       0       1      11       1       2       0

Average        0       1      67       1       2       0
```

Checking Overall System Performance with `sar -A`

The `-A` option provides a view of overall system performance. Use it to get a more global perspective. If data from more than one time slice is shown, the report includes averages.

An example of `sar -A` output follows:

```
unix sfxbs 4.0 2 3B2    08/22/89

14:28:12    %usr    %sys    %sys    %wio   %idle
           local  remote
14:29:12      22      27       0      18      32
14:30:12       6      24       0      13      57

Average       14      26       0      16      44


14:28:12  device   %busy   avque   r+w/s  blks/s  avwait  avserv

14:29:12 hdsk-0       34    10.8      20      39   170.2    17.4

14:30:12 hdsk-0       24    13.6      13      26   236.4    18.8

Average  hdsk-0       29    12.0      16      32   196.6    17.9

14:28:12 runq-sz %runocc swpq-sz %swpocc
14:29:12     1.2      53
14:30:12     1.3      38

Average      1.2      11

14:28:12 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
14:29:12
   local     0       14     100       6      17      67       0       0
   remote    0        0       0       0       0       0
14:30:12
   local     0       12      99       6      16      65       0       0
   remote    0        0       0       0       0       0

Average
   local     0       13     100       6      17      66       0       0
   remote    0        0       0       0       0       0

14:28:12 swpin/s pswin/s swpot/s pswot/s pswch/s
14:29:12    0.00     0.0    0.00     0.0      22
14:30:12    0.00     0.0    0.00     0.0      12

Average     0.00     0.0    0.00     0.0      17

14:28:12 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
14:29:12
   in          0       0       0            0.00       0       0
```

(continued on next page)

```
   out         0        0        0               0.00        0        0
   local      17        2        2       0.28    0.28     2527     1542
14:30:12
   in          0        0        0               0.00        0        0
   out         0        0        0               0.00        0        0
   local      25        2        1       0.50    0.47     1624      295

Average
   in          0        0        0               0.00        0        0
   out         0        0        0               0.00        0        0
   local      21        2        2       0.39    0.38     2075      918

14:28:12  iget/s namei/s dirbk/s
14:29:12       0       2       1
14:30:12       0       4       1

Average        0       3       1

14:28:12 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
14:29:12       0       1     157       1       3       0
14:30:12       0       2      34       2       2       0

Average        0       1      95       1       3       0

14:28:12 proc-sz ov inod-sz ov file-sz ov lock-sz
14:29:12  28/200  0 297/300  0  63/  0  0   6/100
14:30:12  30/200  0 297/300  0  65/  0  0   6/100


14:28:12   msg/s  sema/s
14:29:12    0.00    0.00
14:30:12    0.00    0.00

Average     0.00    0.00

14:28:12  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
14:29:12    1.17   12.87   12.87    5.67   11.28    1.15
14:30:12    1.67    7.08    7.08    9.12    6.33    0.67

Average     1.42    9.97    9.97    7.40    8.81    0.91

14:28:12  pgout/s ppgout/s pgfree/s pgscan/s %s5ipf
14:29:12     0.00     0.00     0.35     8.18   0.00
14:30:12     0.00     0.00     0.00     0.00   0.00

Average      0.00     0.00     0.18     4.09   0.00
```

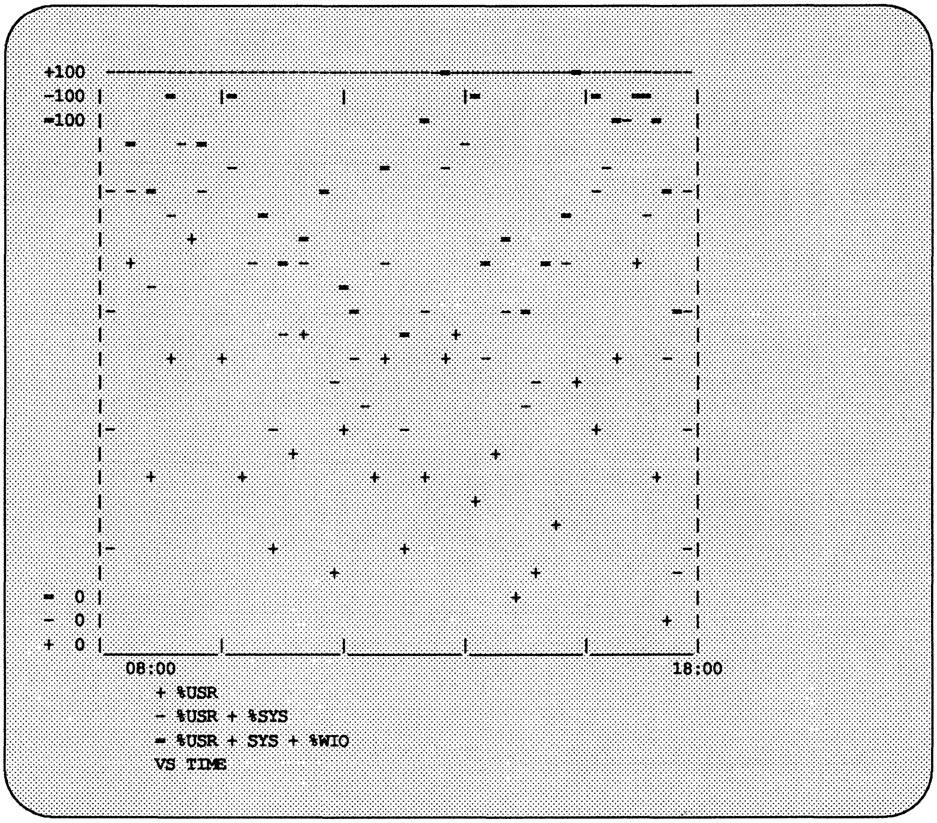
(continued on next page)

```
14:28:12 freemem freeswp
14:29:12     268    3034
14:30:12     351    3009

Average      310    3022

14:28:12 sml_mem   alloc  fail  lg_mem   alloc  fail  ovsz_alloc  fail
14:29:12   95232   73472     0  311296  198656     0      180224     0
14:30:12   95232   75120     0  311296  198656     0      180224     0

Average    95232   74296     0  311296  198656     0      180224     0

14:28:12 open/s create/s lookup/s readdir/s getpage/s putpage/s other/s
14:29:12
  in       0.00     0.00     0.00      0.00      0.00      0.00    0.08
  out      0.00     0.00     0.00      0.00      0.00      0.00    0.00
14:30:12
  in       0.00     0.00     0.00      0.00      0.00      0.00    0.08
  out      0.00     0.00     0.00      0.00      0.00      0.00    0.00

Average
  in       0.00     0.00     0.00      0.00      0.00      0.00    0.08
  out      0.00     0.00     0.00      0.00      0.00      0.00    0.00

14:28:12   snd-inv/s   snd-msg/s   rcv-inv/s   rcv-msg/s dis-bread/s   blk-inv/s
14:29:12         0.0         0.0         0.0         0.0         0.0         0.0
14:30:12         0.0         0.0         0.0         0.0         0.0         0.0

Average          0.0         0.0         0.0         0.0         0.0         0.0

14:28:12 serv/lo - hi    request   request    server    server
               3 -  6    %busy   avg lgth   %avail  avg avail
14:29:12         0          0.0         0      0.0          0
14:30:12         0          0.0         0      0.0          0

Average          0          0.0         0      0.0          0

14:28:12 evpoll/s evpost/s evtrap/s
14:29:12     0.00     0.00     0.00
14:30:12     0.00     0.00     0.00

Average      0.00     0.00     0.00
```

Displaying Collected System Activity Data

sag graphically displays the system activity data stored in a binary data file created by a sar run. Any sar data items may be plotted separately. sag invokes sar and matches strings in the data column header. The next figure shows a typical sag display. Run sar to see what data are available.

Figure 8-1: Example of sag Output



In this figure, the processor is completely utilized over three time intervals: 9–10 A.M., 1–2 P.M., and 3:30–5:30 P.M.. Remember the actual fraction of time that the processor is busy is the sum of user (`%usr`) mode time and system (`%sys`) mode time. When this sum approaches 100 percent, the processor is running at its maximum capacity as configured. The sum of `%usr + %sys + %wio` is about the same as the sum of `%usr + %sys` (`%wio` is low). This means that the disk subsystem is able to handle all requests that the processor generates with little delay. As this example demonstrates, the best way to start, when trying to reduce any slowdown, is reducing processor load.

The `sag` command is useful only if you have a standard output device that can read plotting instructions. Refer to your computer documentation to see if your console terminal has this ability.

Reporting Application Turnaround with `timex`

The `timex` command records the amount of time taken by a command to execute, 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. A record of system consumption can be collected for each application program and then used for tuning the heavily loaded resources. In the following example, the `date` command is used.

```
$ timex -s date
Tue Aug 22 15:07:09 EDT 1989
real        0.17
user        0.00
sys         0.13


unix sfxbs 4.0 2 3B2    08/22/89

15:07:09    %usr    %sys    %sys    %wio   %idle
                   local  remote
15:07:09       8      90       0       2       0


15:07:09 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrit/s
15:07:09
   local     0       4     100       1       1       0       0       0
   remote    0       0       0       0       0       0

15:07:09  device   %busy   avque   r+w/s  blks/s  avwait  avserv

15:07:09 hdsk-0         2     1.0       1       2     0.0    20.0

15:07:09 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
15:07:09       0       0      31       0       1       0

15:07:09 scall/s sread/s swrit/s  fork/s  exec/s rchar/s wchar/s
15:07:09
   in          0       0       0            0.00       0       0
   out         0       0       0            0.00       0       0
   local     157      23       2    3.23    3.23   67918     775

15:07:09 swpin/s pswin/s swpot/s pswot/s pswch/s
15:07:09    0.00     0.0    0.00     0.0      14

15:07:09  iget/s namei/s dirbk/s
15:07:09       0      23       0

15:07:09 runq-sz %runocc swpq-sz %swpocc
15:07:09     1.0     100

15:07:09 proc-sz ov inod-sz ov file-sz ov lock-sz
15:07:09  28/200  0 300/300  0  61/  0  0   6/100

15:07:09   msg/s  sema/s
15:07:09    0.00    0.00
```

(continued on next page)

```
15:07:09  atch/s  pgin/s ppgin/s  pflt/s  vflt/s slock/s
15:07:09   49.46    0.00    0.00   24.73   44.09    2.15

15:07:09  pgout/s ppgout/s pgfree/s pgscan/s %s5ipf
15:07:09     0.00     0.00     0.00     0.00   0.00

15:07:09 sml_mem   alloc  fail  lg_mem   alloc  fail  ovsz_alloc  fail
15:07:09   95232   73344     0  311296  200704     0      180224     0

15:07:09 freemem freeswp
15:07:09     428    3044

15:07:09 open/s create/s lookup/s readdir/s getpage/s putpage/s other/s
15:07:09
  in        0.00     0.00     0.00      0.00      0.00      0.00    0.00
  out       0.00     0.00     0.00      0.00      0.00      0.00    0.00

15:07:09   snd-inv/s   snd-msg/s   rcv-inv/s   rcv-msg/s dis-bread/s   blk-inv/s
15:07:09         0.0         0.0         0.0         0.0         0.0         0.0

15:07:09 serv/lo - hi   request   request    server    server
               3 -  6     %busy  avg lgth    %avail  avg avail
15:07:09         0          0.0         0       0.0          0
```

While date, for its simplicity, was used for the preceding demonstration, it is not representative of many commands because most commands use more system resources.

timex can be used in the following way:

```
timex -s application_program
```

Your application program will operate normally. When you finish running your application and exit, the timex result will be printed on your screen. This can be extremely interesting; you get a precise record of the system resources used while your program was executing.

If accounting is installed and running, then timex can present the information it collects as well. The following options can be used with the existence of accounting:

```
timex [-p[-fhkmrt]] [-o] application_program
```

See timex(1) for more information.

Reporting Location and Seek Distance with sadp

The sadp command has the following format:

```
sadp [-th] [-d device[-drive]] s [n]
```

sadp reports disk access locations and seek distance in tabular (-t) or histogram (-h) form. If neither option is designated, the two reports will be in tabular form.

Valid names for *device* are hdsk for non-SCSI integral disks, sdsk for SCSI integral disks, and fdsk for an integral floppy disk.

drive specifies the disk drives and it may be a single drive number, two numbers separated by a dash to show an inclusive range, or a list of drive numbers separated by commas.

Disk activity is sampled once every second during a specified interval of length *n*. Cylinder usage and seek distance are recorded in units of 20 cylinders. The s option specifies the duration of the sampling interval in seconds. The sampling interval must be 10 seconds or greater. The *n* argument specifies the number of reports to be generated during the sampling interval. The default of *n* is 1.

An example of sadp output for hard disk drive 0 follows.

Figure 8-2: Output from sadp—Cylinder Access Histogram

```
$ sadp -h -d hdsk-0 3600<CR>
CYLINDER ACCESS HISTOGRAM
disk-0:
Total transfers = 33291

12% -|
     |
     |                                                       *
     |                                                       *
     |                                                       *
10% -|                                                       *
     |                                                       *
     |                                                       *
     |                                                       *    *
     |                                                       *    *
 8% -|                                                       *    *
     |                                                       *    *
     |                                                      **    *
     |                                                      **    *
     |                                                      **    * *
 6% -|                                                      **    * *
     |                                                      **    * *   *
     |                                                      **    * *   *
     |                                                      **    * *   *
     |                                                      **    * *   *
 4% -|                                                      **    * *   *
     |                                                      **    * *   **
     |                                                      **    ****  **  *
     |                                                      ***   **** ***  *
     |                                                      ***   ********* *  * *
 2% -|                            *                         ***  ************  * *
     |                            *                         ***  *****************
     |*                           *                         ***  *****************
     |*                           *                         **** *****************
     |**                          *                         **** *****************
     |**.....*................. ..**...*............*.......**********************
  0%-+-----------------------------------------------------------------------------
                          Cylinder number, granularity=8
      081 8     1        2         2        3         4      4         5
        6 0     4        0         7        3         0      6         2
                4        8         2        6         0      4         8
```

Figure 8-3: Output from sadp: Seek Distance Histogram

```
SEEK DISTANCE HISTOGRAM
disk-0:
Total seeks = 30308
      |
20% -|
      |
      |
      |
      |
      |
      | *
      | *
16% -| *
      | *
      | *
      | *
      |**
      |**
      |**
      |**
12% -|**
      |**
      |**
      |**
      |**
      |**
      |**
      |**
 8% -|**
      |**
      |**
      |**
      |***
      |***
      |***   *
      |**** **
 4% -|*********
      |*********
      |*********
      |************
      |**************
      |*****************
      |**********************
      |***************************...*...............................*..............
```

(continued on next page)

Figure 8-3: Output from sadp: Seek Distance Histogram (continued)

```
0% -+-----------------------------------------------------------------------
     =<< <        <     <     <     <     <     <     <
     081 8        1     2     2     3     4     4     5
       6 0        4     0     7     3     0     6     2
                  4     8     2     6     0     4     8
```

Using the sadp output, along with the output of /sbin/mount, or /usr/sbin/prtvtoc, and a table of disk sections (see Appendix A for the default partitioning of your disk(s)), you can identify the file systems with a large amount of I/O activity. In general, try to move files with high activity close together. This will reduce the number of seeks over large distances.

The first graph (Figure 8-2) shows proper block partitioning. This partitioning allows the block to be referenced by the disk head (for reading or writing), within a small region of the disk. In the example, most references (as shown by the percentage of times referenced) point to files near cylinders 450 to 600; a few references point to files around cylinder 250. There are a few references to other files on the disk, but they appear only a small percentage of the time. This graph shows, then, that the most-often used files are grouped together in the same general region of cylinders on the disk; the more clustered the stars on the histogram, the better. Another way to say this is that the disk has an excellent file system configuration.

The second graph (Figure 8-3) shows another aspect of an excellent file system configuration: the head seek distance. This refers to the distance the disk head has to move from the current cylinder to the cylinder of the next block referenced. In the example, most physical seeks were under ten cylinders. Specifically, some 14 percent of the seeks occurred within a distance of 0 to 8 cylinders, and some 17 percent of the seeks occurred within a distance of 8 to 16 cylinders. This means that for approximately one-third of the disk activity, the disk head was forced to move no more than 16 cylinders to reference a given block, and the further left the stars are grouped, the better.

These two graphs show how finely you can tune your system. If, after a working period of weeks or months, you can identify which file systems are consistently the most active, you might consider repartitioning your disks to achieve the maximum from disk access activity (see Chapter 15, "Storage Device Management," and Chapter 5, "File System Administration," for more information).

Samples of Performance Management Procedures

This section describes typical approaches to performance management. First, it describes a general procedure for troubleshooting performance problems. Then, it provides a procedure for reconfiguring the system and shows a sample of a typical system reconfiguration. Finally, it provides a procedure for recovering after an unsuccessful attempt at reconfiguring the system.

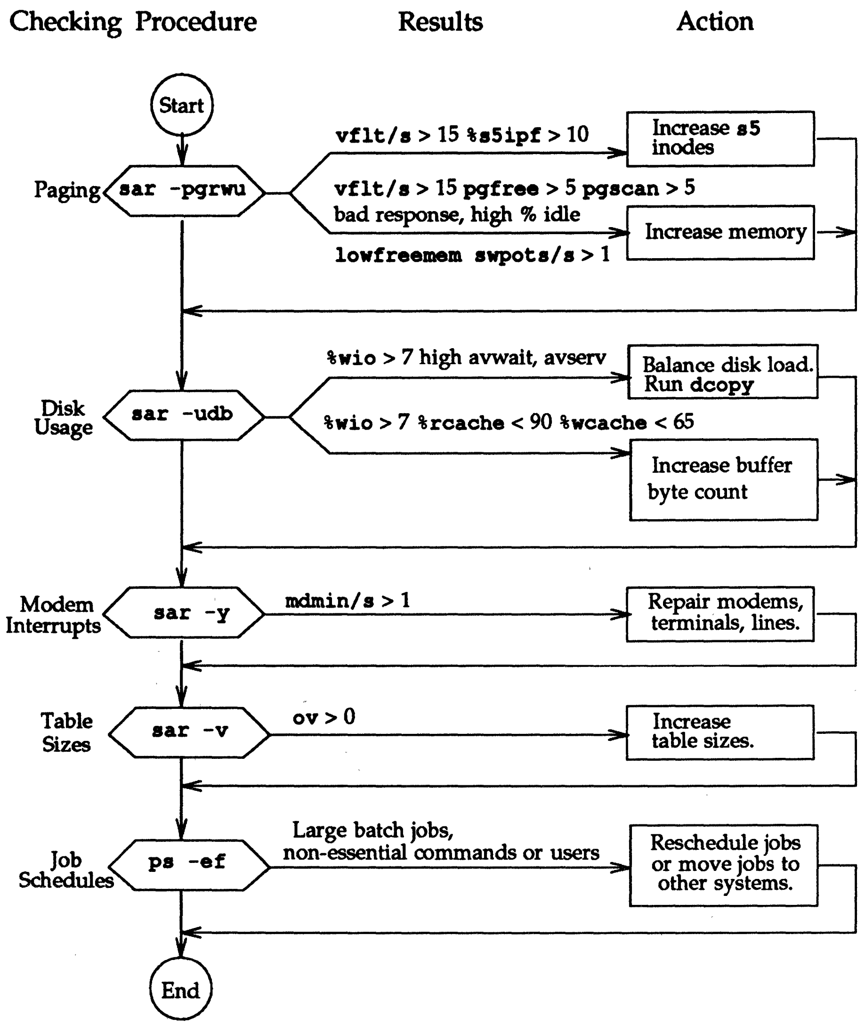
In this section, references are made to tunable configuration parameters. Refer to the the section "Tunable Parameters." for the default values of these parameters and complete instructions for altering them.

Investigating Performance Problems

Locating the source of the problem can require some careful detective work. Hence, the following is not a canned procedure, but a sample approach. It covers basic areas where problems usually surface, and suggests some actions that will alleviate the problem. The most common indication that a problem exists is consistently poor response time. If you have identified a familiar problem area and you need to make changes to your system parameters, see the section "Configuring the UNIX Operating System" in this chapter.

The following figure is an outline of the general approach to troubleshooting:

Figure 8-4: Outline of Typical Troubleshooting Procedure


Checking Procedure
Results
Action
Start
Paging
sar -pgrwu
vflt/s > 15 %s5ipf > 10
Increase s5 inodes
vflt/s > 15 pgfree > 5 pgscan > 5
bad response, high % idle
lowfreemem swpots/s > 1
Increase memory
Disk Usage
sar -udb
%wio > 7 high avwait, avserv
Balance disk load. Run dcopy
%wio > 7 %rcache < 90 %wcache < 65
Increase buffer byte count
Modem Interrupts
sar -y
mdmin/s > 1
Repair modems, terminals, lines.
Table Sizes
sar -v
ov > 0
Increase table sizes.
Job Schedules
ps -ef
Large batch jobs, non-essential commands or users
Reschedule jobs or move jobs to other systems.
End

Checking for Excess Swapping

The first thing to look at is the paging activity, since the paging-in and -out of pages is costly in both disk and CPU overhead. Get the `sar -pgrwu` report. By using this information you can reasonably determine whether more memory is needed or if the excess paging activity is due to too few inodes which indirectly causes reusable pages to be discarded.

If the `vflt/s` value shown by `sar -p` is greater than 15, then look at `sar -g`. High values (greater than 5) for `pgscan/s` and `pgfree/s` imply that the page-stealing daemon is working overtime to find free pages because of a memory shortage. This can be verified by looking at the `ps -elf` report which gives the number of cycles used by the page stealing daemon. The value of `%s5ipf` should also be considered. If it is greater than 10 percent then the freelist of S5 inodes is page-bound which causes reusable pages to be discarded whenever `iget` takes an inode off the freelist. This can be remedied by increasing the tunable `NINODES` found in `/etc/master.d/s5`.

Other indicators of a memory shortage are the `freemem` value of `sar -r` and the `swpot/s` value of `sar -w`. `swpot/s` greater than 1.0 is also an indicator of memory shortage.

If memory shortage occurs frequently then memory should be increased in some way. This can be done in two ways. Uninstalling optional kernel utilities that are not needed by your applications frees the memory used by the utilities so that it can be used by user applications. If this is not possible then extra memory probably needs to be added.

Checking for Disk Slowdowns

If the value of `%wio` (from the `sar -u` report above) is greater than 10 percent, or if the `%busy` for a disk drive (obtained by `sar -d`) is greater than 50 percent, then the system has a disk slowdown. Some ways to alleviate a disk slowdown are:

1. Increase the amount of buffer space.
2. Organize the file system to minimize disk activity. If you have two disks, distribute the file systems for a more balanced load.

3. Consider adding more memory if the situation persists. Additional memory reduces swapping/paging traffic and allows pages to remain in memory (reducing the number of user-level reads and writes that need to go out to disk).
4. Consider adding an additional disk and balancing the most active file systems across the two disks.
5. Consider increasing the logical block size of S5 file systems with lots of large files or even changing the file system type. See the section "Logical Block Size."

Checking for Modem Interrupts

Run `sar -y` to get a report describing activity on terminal devices. If the number of modem interrupts per second, `mdmin/s`, is much greater than 0, your system may have faulty communications hardware.

Checking for Table Overflows

To check for potential table overflows, get the `sar -v` report. This report will let you know if overflows have occurred in the process or inode tables. Overflows in these tables are avoided by increasing `NPROC` and `NINODES` in the `etc/master.d/kernel` and `/etc/master.d/s5` files.

Shifting the Workload to Off-Peak Hours

Examine the files in `/var/spool/cron/crontabs` 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, noninteractive commands (such as `nroff` or `troff`) at off-peak hours. You may also want to run such commands with a low priority by using the `nice` or `batch` shell commands.

Configuring the UNIX Operating System

During the boot procedure, the boot program reads from disk a program called `unix`, loads it into memory, and executes it. This file, `unix` (often referred to as the bootable operating system), defines the running UNIX system on your machine.

The absolute path name of this file is `/stand/unix`.

NOTE `/unix` is symbolically linked to `/stand/unix` for compatibility with earlier releases.

For simplicity, we will refer to this file as the bootable operating system, or `unix`.

The `unix` that runs on your machine is specifically configured for the hardware and software currently on your machine.

All software changes that need to be incorporated into the bootable operating system are specified in the `/stand/system` file, or in one of the `/etc/master.d` files. For simplicity, we refer to `/stand/system` as the `system` file, and to files found in `/etc/master.d` as `master` files.

There are two reasons that the bootable operating system would need to be reconfigured:

- software changes were made to the system that need to be incorporated into the bootable operating system
- hardware resources were added to or removed from the system

The software changes that could be made to a system include the following:

- changing the definition of a driver in the `/etc/master.d` directory
- changing tunable parameters found in a `master` file
- adding or deleting a driver or module definition in a `master` file
- making changes to the `system` file
- adding or removing driver modules from `/boot`

While the system file is read directly by the configuration software, changes to master files require execution of the mkboot(1M) command before the configuration process is begun. First, you must change the appropriate file(s) in the directory /etc/master.d, and then execute a separate mkboot command for each changed master file. The mkboot command will read the master file and create a new object file in the /boot directory.

The /boot directory contains object files for use in the configuration process; these object files contain configuration information necessary for various hardware and software drivers on your system. The system file is used to specify which of the object files in /boot must be configured into unix, and which are to be excluded.

Some modules in /boot are included in the bootable operating system even if they are not explicitly included in the system file. If you want to explicitly exclude a driver in /boot, it is best to do so with an EXCLUDE statement in /stand/system (see system(4)). Do not move the driver to a new file name in /boot or some other directory to exclude it from the bootable operating system.

Hardware changes always require the configuration of a new unix. These changes include adding or removing a board, a tape drive, and so on.

One topic we have not considered is the installation of a new software package. Usually, the installation procedure for installing a new software package that requires a modification of the bootable operating system includes some modification of the system file and drivers in /boot; this is usually done directly by the installation software provided with the software package. The system is then rebooted at the end of the installation process, and a new bootable operating system is built, loaded, and executed.

Standard procedures for installing new hardware are given in the documentation that accompanied your computer.

Configuration Scenarios

The configuration of a new `unix` can occur in one of five ways:

- on powerup, the system detects that the `system` file is newer than the `unix` file
- a `shutdown -i6` or `init 6` is executed at the shell prompt and the `system` file is newer than `unix`
- the name of the `system` file (or another text file) is entered at the firmware prompt
- the `cunix` command is executed at the shell prompt
- a hardware change is made that makes the current `unix` inoperable

Of all the above methods, using the `cunix` command provides the most flexibility, and has the added advantages of allowing you to build a new bootable operating system without rebooting the machine, and without overwriting the currently running bootable operating system. The `cunix` command is particularly useful in development environments where the possibility exists of creating a bootable operating system that will not execute properly, or not at all, because of bad changes made to the `system` file, a bad change made to a `master` file, or symbol-referencing problems.

The remaining sections in this chapter describe configuring a new operating system through a reboot, recovering from an attempt to boot an unbootable operating system, and the use of `cunix`. A procedure for configuring a new `mUNIX` (a version of the UNIX operating system that runs during the configuration process) is also given.

Reconfiguring the System Through a Reboot

This section tells you how to configure a new bootable operating system through a system reboot. The procedure includes the modification of system tunable parameters as an example, though any change in the hardware or software configuration of your system could be substituted for that part of the procedure.

There are four things to remember when reconfiguring the operating system:

1. Always copy the existing bootable operating system (`/stand/unix`) to another file; it is recommended that you copy it to a file in the root directory and not to another file in the `/stand` directory. This way, in the event you create an unbootable `unix`, you can boot `mUNIX` from firmware and copy the existing bootable operating system from the root directory back to `/stand/unix`.
2. For each driver or module that you modified in `/etc/master.d`, execute a separate `mkboot` command.
3. When reconfiguring the operating system do not arbitrarily change the node name (`NODE`) of the computer. If basic networking has been established, a change in node name must be coordinated with all interfacing systems.
4. Take detailed notes of everything you do so that you can identify and correct any mistakes you may make. In the event that you need to contact a service representative for your computer, your notes will be an invaluable aid in resolving your problem.

There are two major steps in reconfiguring the operating system:

1. change the system configuration (in this example, modify tunable parameters)
2. rebuild the operating system

These steps are described in the procedures below.

Modifying the Tunable Parameters

Step 1: Log in as `root`.

Step 2: Copy the existing `/stand/unix` to `/oldunix`.

```
# cp /stand/unix /oldunix
```

Step 3: Change the present working directory to `/etc/master.d`.

```
# cd /etc/master.d
```

Step 4: Edit the applicable `master` files to modify the tunable parameters. Keep a record of all changes you make; this is important in case you make a mistake that results in the configuration of an unbootable operating system or the failure of the configuration process.

Configuring a New Bootable Operating System

Step 5: Change the present working directory to `/boot`.

```
# cd /boot
```

Step 6: Execute the `mkboot` command to create a bootable object file for each of the files modified in `/etc/master.d`. For example, if some of the tunable parameters in the `/etc/master.d/kernel` were modified, you would enter:

```
# /usr/sbin/mkboot -k KERNEL
```

If the tunables that you changed were in another file (for example, `/etc/master.d/sem`), the `mkboot` command does not take the `-k` option:

```
# /usr/sbin/mkboot SEM
```

Step 7: Execute the `cunix` command to make a new unix:

```
# cunix -o /newunix
```

If `cunix` works successfully, move `newunix` to `/stand/unix`:

```
# mv /newunix /stand/unix
```

and reboot, otherwise fix the problems and try again. (For more information in `cunix`, see `cunix`(1M).)

NOTE Using the `touch` command on the file `stand/system` and then rebooting will also remake your system, as in previous releases, but will not allow you to check for errors before you shut down.

See "Sample System Configuration" in the next section for an example of the prompts that appear during the reboot and configuration process.

Step 8: Reboot the system:

```
# shutdown -i6
```

If the system will not boot, you must boot /stand/mUNIX to bring up a system so that you can repair problems.

If you cannot determine and/or fix the problem, you must undo the changes made to the files in /etc/master.d in Step 4; then, repeat Steps 5 and 6. After completing Step 6, move /oldunix back to /stand/unix and reboot.

Sample System Reconfiguration

The following is an illustration of a typical scenario for reconfiguring an AT&T 3B2 Computer because of adding more memory.

Because of the additional memory, many tunable parameters should be increased. Most of them are in the /etc/master.d/kernel file. The command line entries and system responses in the illustration below show the reconfiguration and rebooting of the operating system to support these new parameters. The illustration also indicates that tunable parameters for semaphores are being modified. The editing of the /etc/master.d/kernel and /etc/master.d/sem files is not shown.

```
# cp /stand/unix /oldunix
# cd /etc/master.d
# ed kernel
```

NOTE: Editing of `/etc/master.d/kernel` *is not shown.*

```
q
# cd /boot
# mkboot -k KERNEL
# cd /etc/master.d
# ed sem
```

NOTE: Editing of `/etc/master.d/sem` *is not shown.*

```
q
# cd /boot
# mkboot SEM
```

NOTE: If parameters in other `/etc/master.d` *files are changed, execute* mkboot *on the uppercase name for each changed file. Only the KERNEL file requires the* `-k` *option. See* mkboot*(1M).*

```
# cd
# touch /stand/system
# shutdown -i6
```

NOTE: A series of messages are displayed ending with the following:

```
INIT: New Run level: 6
The system is coming down. Please wait.
System services are now being stopped.
A new unix is being built

/stand/unix is being created

CONFIGURATION SUMMARY
_____________________
     -----driver----- #devices major
     LOG                  1      50
     CLONE                1      63
     PRF                  1      49
     SXT                  1      48
     GENTTY               1      20
     PORTS                2       1,   2
     MEM                  1      18
     IUART                1       0
     IDISK                1      17
     HDELOG               1      16

     -----module-----
```

(continued on next page)

```
     INTP
     ELF
     COFF
     FIFOFS
     BFS
     .
     .
     .
     ----device info----
                  major            minor
     rootdev        17               0
     swapdev        17               1
```

NOTE: A series of messages is displayed ending with the following:

```
The system is down.

SELF-CHECK

UNIX System V Release 4.0 AT&T 3B2 Version 2
Node marmaduk
Total real memory  = 2087157
Available memory   = 1236992
********************************************************

Copyright (c) 1984, 1986, 1987, 1988, 1989 AT&T  - All Rights Reserved

THIS IS UNPUBLISHED PROPRIETARY SOURCE CODE OF AT&T INC.
The copyright notice above does not evidence any actual or
intended publication of such source code.

********************************************************
The system is coming up.  Please wait.

AT&T 3B2 SYSTEM CONFIGURATION:

Memory size: 2 Megabytes
System Peripherals:
        Device Name        Subdevices          Extended Subdevices
        SBD
                           Floppy Disk
                           30 Megabyte Disk
                           72 Megabyte Disk
        PORTS
        MAU
The system is ready.

Console Login:
```

Recovering from an Unbootable Operating System

If your attempt at configuring a new bootable operating system is unsuccessful, resulting in an unbootable operating system or the failure of the configuration process, you can get a viable version of the system running using the procedure outlined below. This procedure can also be used if you configure a new bootable operating system that performs poorly and you want to recover the previously used /stand/unix.

It is assumed that you previously saved a copy of /stand/unix as /oldunix.

Step 1: If the system is in the firmware state, skip to Step 2.

If you are in single- or multi-user mode, bring the system to the firmware state:

```
# shutdown -i5
```

Note that if you are in multi-user mode, you must be logged in as root.

If you are in the shell spawned during an error in the configuration process, use exit or CTRL-d to go to firmware mode.

If for some reason the system is not able to come up at all, see the hardware documentation for your computer for instructions on getting the system to firmware mode in this case (most computers are equipped with some form of hardware reset switch).

Step 2: Enter the firmware password. When prompted, enter /stand/mUNIX:

```
Enter name of program to execute [ ]: /stand/mUNIX
```

The system boots mUNIX, a version of the operating system that always resides in /stand.

Step 3: After the system has rebooted and you have logged in as root, move /oldunix back to /stand/unix:

```
# mv /oldunix /stand/unix
```

This returns the previous working version of the operating system to /stand/unix.

Step 4: If you made any changes to files in `/etc/master.d` or to the `/stand/system` file before a configuration attempt that failed, undo all the changes made at this time (if you have not done so already) so that these system files match the current bootable operating system in `/stand/unix`. If you had saved copies of these files before changing them, simply move the old files back to `/etc/master.d`. Then, execute a separate `mkboot` for each module corresponding to the `master` files as shown in the section "Configuring a New Bootable Operating System."

Step 5: Reboot the system:

```
    # shutdown -i6
```

Step 6: When the `Console Login:` prompt appears you can log in to your system.

Try to determine what you did to cause the configuration process to fail so that you can avoid the problem in the future. If you are unable to determine what went wrong, and repeated attempts at configuring a new operating system fail, retain all notes and other documentation that you kept during the attempt(s) (including changes made to `master` files and/or the `/stand/system` file), and contact your computer service representative.

User-Level Configuration of the UNIX System

Configuring `unix` at the user-level has several major advantages over configuring `unix` automatically through a reboot:

- no reboot of the system is necessary (i.e., system remains available to all users)
- the reconfigured operating system can be placed in a file other than `/stand/unix`, leaving the current bootable operating system intact
- an alternate `system` file and `/boot` directory can be specified

Typically, the `cunix` command will be used as in the following example:

```
    cunix [-f system] [-o new_unix]
```

The `-f` option specifies the pathname of the `system` file to be used for the configuration of the new bootable operating system. By default, this is /stand/system, but can be any text file in system(4) format.

There are also other options to cunix that allow you to further customize your configuration environment. See the cunix(1M) manual page for more information.

Configuring a New mUNIX

As previously described, mUNIX is a version of the UNIX system that runs during the configuration process. It was configured originally using the /stand/mini_system file as the system file.

You may want to configure a new mUNIX to account for your particular operating environment. To do this, just make changes to the /stand/mini_system file as necessary and execute the cunix command, as in the following example:

```
cunix -f /stand/mini_system -o new_mUNIX
```

This command will build new_mUNIX (in the current directory) using the mini_system file as the system file. Do not make any changes to master files unless you also intend to reconfigure unix along with mUNIX.

Once the configuration process is complete, you should move the new mUNIX to /stand, optionally saving the old one before doing so, as in the following:

```
mv /stand/mUNIX old_mini_unix
mv new_mUNIX /stand/mini_unix
```

This causes your newly configured mUNIX to be the one used for future configurations of the operating system, and retains a copy of the old mUNIX in the current directory. Of course, you can keep an old copy of mUNIX in any directory, including the root directory.

Tunable Parameters

Tunable system parameters are used to set various table sizes and system thresholds to handle the expected system load. Caution should be used when changing these variables since such changes can directly affect system performance. For the most part, the initial tunable parameter values for a new 3B2 computer are acceptable for most configurations and applications. If your application has special performance needs, you may have to experiment with different combinations of parameter values to find an optimal set.

Note that whenever a parameter's value is being changed from the default value to a much higher value, you should read the `master.d` file to determine the type of parameter data that affects its maximum value.

The tunables for the core package can be found in the following `/etc/master.d` files delivered with the core package:

`kernel`	Kernel Tunables
`hrt`	High Resolution Timers
`ports`	Ports Board--STREAMS Version
`log`	STREAMS Log Driver
`sad`	STREAMS Administrative Driver
`ts`	Time Sharing Scheduler
`s5`	System V File System Type

Figure 8-5 shows the default values for the tunable parameters found in these files for systems equipped with 4 megabytes of random access memory (RAM).

Also included in Figure 8-5 are the default values for the tunable parameters found in the in the `/etc/master.d` files of the following packages:

UFS Utilities

`/etc/master.d/ufs`	4.2BSD Fast File System

System Performance Analysis Utilities

`/etc/master.d/prf`	Kernel Profiler

Interprocess Communication Utilities

`/etc/master.d/msg`	Messages
`/etc/master.d/shm`	Shared Memory
`/etc/master.d/sem`	Semaphores

The following packages also have tunable parameters found in `/etc/master.d` files. The files associated with the package and where the respective tunables are found are also given.

Internet Utilities

`/etc/master.d/arp`	Address Resolution Protocol
`/etc/master.d/ip`	Internet Protocol
`/etc/master.d/tcp`	Transmission Control Protocol
`/etc/master.d/udp`	User Datagram Protocol
`/etc/master.d/llcloop`	Loopback Driver

Found in: *Network User's and Administrator's Guide*

Network File System Utilities

`/etc/master.d/nfs`	Network File System

Found in *Network User's and Administrator's Guide*

Remote File Sharing Utilities

`/etc/master.d/rfs`	Remote File Sharing

Found in: *Network User's and Administrator's Guide*

XENIX Compatibility Package

`/etc/master.d/xnamfs`	XENIX Semaphores

Found in: *BSD/XENIX® Compatibility Guide*

The following notes apply to Figure 8-5:

- The parameters are set to specific values, as defined in the appropriate `/etc/master.d` file. The default value and the size in bytes for each entry are shown in the figure.
- A dash (–) is used in the size information to indicate parameters that do not affect the size of the kernel when the values are changed. These parameters instead act as flags, limits, or provide a naming function.

Figure 8-5: Suggested Parameter Values

master.d File	Tunable Type	Parameter	Default Value	Size per Entry in Bytes
kernel	Gen. Kernel Tunables	NCALL	60	16
		ARG_MAX	5120	-
		FLCKREC	300	-
		MAXUP	25	-
		NCLIST	0	-
		NPROC	200	212
		PUTBFSZ	2000	1
		ROOTFSTYPE	"s5"	-
	System Information	SYS	"UNIX Sys. V"	-
		NODE	"unix"	-
		REL	"4.0"	-
		VER	"2"	-
		SRPC_DOMAIN	""	-
	Hardware Information	ARCHITECTURE	"M32100"	-
		HW_SERIAL	""	-
		HW_PROVIDER	"AT&T"	-
	Buffer Cache	NBUF	100	88
		NHBUF	64	12
		NPBUF	20	52
		BUFHWM	200	-
	Paging	FSFLUSHR	1	-
		NAUTOUP	60	-
		SPTMAP	100	8
		MAXPMEM	0	

Figure 8-5: Suggested Parameter Values (continued)

master.d File	Tunable Type	Parameter	Default Value	Size per Entry in Bytes
		GPGSLO	25	-
		MINARMEM	25	-
		MINASMEM	25	-
	Per Process Limits	SHLBMAX	2	-
		SCPULIM	0x7fffffff	-
		HCPULIM	0x7fffffff	-
		SFSZLIM	0x100000	-
		HFSZLIM	0x100000	-
		SDATLIM	0x1000000	-
		HDATLIM	0x1000000	-
		SSTKLIM	0x1000000	-
		HSTKLIM	0x1000000	-
		SCORLIM	0x100000	-
		HCORLIM	0x1000000	-
		SFNOLIM	0x18	-
		HFNOLIM	0x400	-
		SVMMLIM	0x1000000	-
		HVMMLIM	0x1000000	-
	File Access Features	RSTCHOWN	0	-
		NGROUPS_MAX	16	-
	STREAMS	NSTRPUSH	9	-
		STRCTLSZ	1024	-
		STRMSGSZ	0	-
		STRTHRESH	2000000	-
	Scheduler Information	MAXCLSYSPRI	99	-

Figure 8-5: Suggested Parameter Values (continued)

master.d File	Tunable Type	Parameter	Default Value	Size per Entry in Bytes
		SYS_NAME	"SYS"	-
		INITCLASS	"TS"	-
	XENIX Shared Data	XSDSEGS	25	12
		XSDSLOTS	3	20*XSDSEGS
hrt	High Resolution Timers	HRTIME	50	60
		HRVTIME	50	60
ports	Ports Board	SAVEXP	125	1
log	STREAMS Logging	NLOG	16	12
sad	STREAMS Admin. Driver	NSTRPHASH	64	4
		NAUTOPUSH	32	44
ts	Time Sharing	TSMAXUPRI	20	-
s5	S5 File Type	NINODE	400	132
ufs	Fast File System	UFSNINODE	150	268
		NDQUOT	200	60
prf	Profiler	PRFMAX	2048	1
msg	Message	MSGMAP	100	8
		MSGMAX	2048	-
		MSGMNB	4096	-
		MSGMNI	50	53
		MSGSSZ	8	1024
		MSGTQL	40	12
		MSGSEG	1024	8

Figure 8-5: Suggested Parameter Values (continued)

master.d File	Tunable Type	Parameter	Default Value	Size per Entry in Bytes
sem	Semaphores	NBPW	4	-
		SEMMAP	10	8
		SEMMNI	10	34
		SEMMNS	60	12
		SEMMNU	30	8x(SEMUME+2)
		SEMMSL	25	-
		SEMOPM	10	8
		SEMUME	10	8x(SEMMNU)
		SEMVMX	32767	-
		SEMAEM	16384	-
shm	Shared Memory	SHMMAX	131072	-
		SHMMIN	1	-
		SHMMNI	100	112
		SHMSEG	6	12xNPROC

Kernel Tunables

General Kernel Tunables

The following general kernel parameters are defined in the `/etc/master.d/kernel` file.

`ARG_MAX` This is the maximum number of characters (including NULL characters) allowed in the argument and environment strings passed to an `exec` system call. This can be increased to allow larger argument lists, but it should not be less than 5120. If it is increased, it should be no more than about an eighth of `SSTKLIM` (see "Per Process Limits" below) so that there is room for both the pointer arrays and the ordinary stack frames.

`FLCKREC`	Specifies the number of records that can be locked by the system. The default is 300.
`MAXUP`	Specifies how many concurrent processes a non-superuser is allowed to run. The entry is normally in the range of 15 to 25. This value should not exceed the value of `NPROC` (`NPROC` should be at least 10 percent more than `MAXUP`). This value is per user identification number, not per terminal. For example, if twelve people are logged in with the same user identification, the default limit would be reached very quickly.
`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 greater than 2 and is normally in the range of 10 to 70. The default value is 60. 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 outputs the following message on the system console: `PANIC: Timeout table overflow`
`NCLIST`	`NCLIST` is no longer used by 3B2 UNIX systems. Other machine architectures may use `clist`, however.
`NPROC`	Specifies how many process table entries to allocate. Each table entry represents an active process. The swapper is always the first entry and `/sbin/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`(2) system call returns the error `EAGAIN`. The default value of `NPROC` is 200. It should probably be no less than 50.

PUTBUFSZ The size, in bytes, of the circular buffer putbuf used to record various system messages, including PANIC messages. The messages in putbuf can be seen in a system dump by using the crash command to print the contents of putbuf as ASCII characters. There should be no reason to change this value unless you are doing operating system development.

ROOTFSTYPE Specifies the file system type of the root file system. This is used to determine the format of the file system.

System Information

The following system information tunables are defined in the /etc/master.d/kernel file.

SYS Specifies the system name. The default system name is UNIX_System_V (see the procedure "Changing the System Name and Node Name" in Chapter 16, "System Setup").

NODE Specifies the node name of the system. The default node name is unix (see the procedure "Changing the System Name and Node Name" in Chapter 16, "System Setup."

REL Specifies the UNIX system release.

VER Specifies the version. This value may be 1 or 2.

SRPC_DOMAIN The name of the "Secure RPC Domain," the realm in which a uid space is unique. When using secure RPC, each user is assigned a "netname" which is of the form

```
Operating System.UserId@Realm
```

So, if the user id of each user is the same for machines A, B, and C, then SRPC_DOMAIN should be set to the same name on A, B, and C. Thus, if the SRPC_DOMAIN name for machines A, B, and C is "documentation," then user 1701 from any one of those machines would have the netname

```
    unix.1701@documentation
```

See "RPC Administration" in the *Programmer's Guide: Networking Interfaces* for more information on secure domain.

Hardware Information

The following parameters are defined in the `/etc/master.d/kernel` file.

`ARCHITECTURE`	The machine architecture information.
`HW_SERIAL`	The serial number of your machine. Its value, of course, must be filled in by the administrator.
`HW_PROVIDER`	The hardware provider's name.

Buffer Cache

The following parameters are defined in the `/etc/master.d/kernel` file.

`NBUF`

Block I/O uses both buffers and buffer headers. Whenever a buffer header is needed, but no free ones are available, the system dynamically allocates more buffer headers in chunks of `NBUF` at a time. There is no limit to the total number of buffer headers in the system, however, the tunable `BUFHWM` limits the number of kilobytes that may be used by buffers. This effectively limits the number of buffer headers that will be allocated.

Once allocated, buffer header space cannot be freed for other uses. Thus, care should be taken when raising the value of `NBUF`. A higher value of `NBUF` will decrease the number of times the Kernel Memory Allocator must be called to allocate space for buffer headers, but this could also result in the allocation of headers that are not used.

`NHBUF`

Specifies how many "hash buckets" 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. `NHBUF` must be specified in `/etc/master.d/kernel`.

`NPBUF`	Specifies how many physical I/O buffers to allocate. One I/O buffer is needed for each physical read or write active. Each entry contains 52 bytes. The default value is 20.
`BUFHWM`	`BUFHWM` limits the number of kilobytes of memory that can be used by block I/O buffers. If `sar -b` shows the buffer hit ratio to be low, then `BUFHWM` and/or `NBUF` should be increased.

Paging

A paging daemon, `pageout`, exists in the system. Its sole responsibility is to free memory as the need arises. It uses a "least recently used" algorithm to approximate process working sets and writes those pages that have not been touched during some period of time out to disk. The page size is 2048 bytes. When memory is exceptionally tight, the working sets of entire processes may be swapped out.

The first two tunables are for file system hardening. The remaining tunable parameters determine how often `pageout` runs and under what conditions. The default values in `/etc/master.d/kernel` should be adequate for most applications.

`FSFLUSHR`	This is the file system flush rate. It specifies the rate in seconds for checking the need to write the file system buffers, modified inodes, and mapped pages to disk. The default is one second. (This has replaced `BDFLUSHR` of previous releases.)
`NAUTOUP`	The `NAUTOUP` entry specifies the buffer age in seconds for automatic file system updates. System buffers and other cached file attributes (such as inodes) are written to the hard disk when they have 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.

`SPTMAP`	The number of map entries for free space accounting for the dynamic portion of the kernel virtual address space. The dynamic kernel space is used by drivers and `kmem_alloc` for execution time allocation of kernel memory. It should never be made smaller. It should be made larger if the following warning message is seen:
	`rmfree map overflow SPTMAPADDR.`
	`Lost N items at LOSTADDRESS.`
	where `SPTMAPADDR` is the hexadecimal address of the `sptmap` array and `LOSTADDRESS` is a kernel virtual address in the dynamic allocation address interval. (On the 3B2, this interval is 0x40300000 <= LOSTADDRESS < 0x40500000.)

`MAXPMEM`	Specifies the maximum amount of physical memory to use in pages. The default value of 0 specifies that all available physical memory be used.
`GPGSLO`	Specifies the low water mark of free memory in pages for `pageout` 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.)
`MINARMEM`	Specifies the minimum number of memory pages reserved for the text and data segments of user processes.
`MINASMEM`	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).
`PAGES_UNLOCK`	Unused.

Per Process Limits

`SHLBMAX`	Specifies the maximum number of shared libraries that can be attached to a process at one time. This applies only to `COFF` shared executables.

The following tunables are soft and hard limit pairs on process resource limits. These limits are given to process 0; thereafter child processes inherit the parent process's hard and soft limits. However, whenever a process **execs** a file whose set-user-id or set-group-id bit has been set, the resource limits of that process are reinitialized to the default system limits.

Processes can change their own values of these limits using **setrlimit** (see `getrlimit`(2)). Soft limits may be changed but must remain less than or equal to the hard limits. Only processes whose effective user ID is equal to 0 (`root`) may raise their hard limits. Any process may lower its hard limit.

A value equal to `RLIMIT_INFINITY` (0x7ffffff on the 3B2) indicates a resource without limitation.

See `getrlimit`(2) for more information on hard and soft limits.

`SCPULIM`	The soft limit of the maximum combined user and system CPU time, in seconds, that a process is allowed. A `SIGXCPU` signal will be sent to processes whose CPU time exceeds this value.
`HCPULIM`	The maximum value of `SCPULIM`.
`SFSZLIM`	The soft limit specifying the largest offset, in bytes, of any single file that may be created by the process. A `SIGXFSX` signal will be sent to processes that attempt to write a file whose offset is greater than this value. In addition, the write will fail with an `EFBIG` error.
`HFSZLIM`	The maximum value of `SFSZLIM`.
`SDATLIM`	The soft limit specifying the maximum size, in bytes, of a process's heap. If a process attempts to extend its heap beyond this limit using `brk`(2), the attempt will fail and `errno` will be set to `ENOMEM`.

`HDATLIM`	The maximum value of `SDATLIM`.
`SSTKLIM`	The soft limit specifying the maximum size, in bytes, of the stack segment for a process. This defines the limit of automatic stack growth by the system. A `SIGSEGV` signal will be sent to processes that attempt to grow the stack beyond this value. Unless the process has arranged to catch this signal on a separate stack (see `signal-stack`(2)) this will terminate the process.
`HSTKLIM`	The maximum value of `SSTKLIM`.
`SCORLIM`	The soft limit specifying the largest size, in bytes, of a core file that may be created. A soft limit of 0 will prevent the creation of core files.
`HCORLIM`	The maximum value of `SCORLIM`.
`SFNOLIM`	The soft limit specifying the maximum number of open files the process may have. When this limit is exceeded, attempts to open files will fail and `errno` will be set to `EMFILE`.
`HFNOLIM`	The maximum value of `SFNOLIM`.
`SVMMLIM`	The soft limit specifying the maximum address space that may be mapped to a process. Attempts to increase a process's address space beyond this value (i.e., `brk`(2), `shmat`(2), `mmap`(2)) will fail with a `ENOMEM` error.
`HVMMLIM`	The maximum value of `SVMMLIM`.

File Access Features

`RSTCHOWN`	`RSTCHOWN` is the restricted file ownership changes flag. Only 0 and 1 are valid values for `RSTCHOWN`. A value of 0 is the System V Release 3 compatibility mode. As in Release 3, the owner of a file can change user ID and group ID of the file to any value, including nonexistent user IDs and group IDs. `RSTCHOWN` set to 1 designates the FIPS/BSD compatibility mode. This restricts the ability to change ownership of the file. Only the superuser or root processes (those whose UID is 0) are able to

	change the ownership of a file. The owner of the file may only change the group ID of the file to one of the groups in which the owner has membership (see `getgroups`(1)). Superuser and root processes may change the group ID of any file to any value. (`RSTCHOWN` set to 1 is FIPS/BSD compatibility mode.)
`NGROUPS_MAX`	Specifies the maximum number of groups in which a process can have membership (see `getgroups`(1)).

STREAMS

The following tunable parameters are associated with STREAMS processing. These parameters are defined in the `/etc/master.d/kernel` file.

`NSTRPUSH`	The maximum number of modules that may be pushed onto a Stream. This is used to prevent 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.
`STRMSGSZ`	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. A value of zero indicates no upper bound. A value of 4096 is sufficient for existing applications.
`STRCTLSZ`	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 recommended value of 1024 is more than sufficient for existing applications.

`STRTHRESH`	The maximum total of bytes streams are normally allowed to allocate. When the threshold is passed, users without the appropriate privilege will not be allowed to open streams, push streams modules, or write to streams devices; they will fail with `ENOSR` (out of streams resources). Users with appropriate privilege will always be allowed to do anything. Note also that the threshold applies to the output side only, thus data coming into the system (for example, the console) is not affected and will continue to work properly. A value of zero means there is no threshold. `STRTHRESH` should be set to about 1/4 to 1/2 of the total system memory. The default of 2000000 (approximately 2 megabytes) is a good maximum for a 4 megabyte system.

Scheduler Information

The following parameters are defined in the `/etc/master.d/kernel` file.

`MAXCLSYSPRI`	Maximum global priority used by the SYS scheduling class for scheduling kernel processes. Changing this changes the range of priorities used to schedule kernel processes and can have a significant effect on the performance of the system. In general, there is no need to change this unless you are adding new scheduling classes or reconfiguring the priorities of other currently configured classes. If it is set to a value below 39, the kernel will automatically set it to 39 at boot time because it needs a range of 40 priorities for the SYS class. (See the "Process Scheduling" chapter for detailed information.)
`SYS_NAME`	The character string name of the system scheduling class. There is no need to change the default unless you are configuring a different scheduling class with the name `SYS`.
`INITCLASS`	Specifies the scheduling class assigned to the `init` process. This class will be inherited by all processes on the system except descendents of a process whose class has been reset using `priocntl`(2). Should not be changed without good reason.

XENIX Shared Data

The following parameters are defined in the `/etc/master.d/kernel` file.

`XSDSEGS`	Specifies the number of shared data segments in the system. The minimum value is 1, and the default and maximum value is 25.
`XSDSLOTS`	(`XSDSEGS` x `XSDSLOTS`) specifies the maximum number of shared data segment attachments allowed in the system. The minimum value of `XSDSLOTS` is 1, and the default and value of `XSDSLOTS` is 1, and the default and maximum value is 3.

High Resolution Timers

The configuration parameters for High Resolution Timers are found in the `/etc/master.d/hrt` file. They are:

`HRTIME`	`HRTIME` is used to define the size of the `hrtimes` array. The `hrtimes` array is used for keeping track of sleep and alarm requests for the standard, real-time clock.
`HRVTIME`	`HRVTIME` is used to define the size of the `itimes` array. The `itimes` array is used for keeping track of the alarm requests for the clocks measuring user process virtual time and a process's virtual time.

Ports Board

The configurable parameter for the Ports Board is found in the `/etc/master.d/ports` file. It is:

`SAVEXP` Number of saved express jobs on the ports and high-ports boards. Should be increased if the message

```
PORTS: EXPRESS QUEUE OVERLOAD:
One entry lost, bin = N, pid = M
```

is printed (where *N* is the board number and *M* is the port number). It probably will not need to be changed.

STREAMS Log Driver

The configurable parameter for the STREAMS log driver is found in the file `/etc/master.d/log`. It is:

`NLOG`	The number of minor devices that are available through the clone interface of the log driver (`/dev/log`). If an open of `/dev/log` fails with `errno` set to `ENXIO`, this number may need to be increased.

STREAMS Administrative Driver

The configurable parameters for the STREAMS Administrative Driver (SAD) are found in the file `/etc/master.d/sad`. They are:

`NSTRPHASH`	The size of the internal hash table. This will probably never need to be changed unless the number of drivers on the system gets very, very large.
`NAUTOPUSH`	The number of devices that can be configured to be auto-pushed. If the `SAD_SAP ioctl` fails with `errno` set to `ENOSR`, then this number should be increased.

Time Sharing Scheduler

The following parameter for the Time Sharing Scheduler is found in the `/etc/master.d/ts` file.

`TSMAXUPRI`	The range within which users may adjust the user priority of a time-sharing process is `-TSMAXUPRI` to `+TSMAX-UPRI`. Configuring higher values gives users more control over the priority of their processes (note that only super-user can raise priority in any case). The default value of 20 provides a degree of control equivalent to

what has been available in the past through the nice(2) interface.

S5 File System Type

The following parameter for the System V File System is found in the /etc/master.d/s5 file.

NINODE	The number of inode entries in the S5 inode table. If sar -v shows that table overflows are occurring or if sar -g shows %s5ipf is greater than 10 percent, then the value should be raised. On the other hand, if sar -v consistently shows that the inode table is underutilized, then the value could be lowered. NINODE should be greater than ncsize which is specified in /etc/master.d/kernel. ncsize determines the number of inodes used by the directory lookup cache. A general guideline for the value of NINODE is 100 S5 inode entries for each megabyte of memory or ncsize + 100, whichever is greater.

Fast File System Type

The following parameters are associated with the Fast File System and are found in the /etc/master.d/ufs file.

UFSNINODE	The number of ufs inode table entries. If the main file system is ufs, then 100 inode entries per megabyte of memory should be allocated and the number should be greater than the value ncsize in /etc/master.d/kernel.
NDQUOT	The size of the kernel quota table. There is one entry for each user, thus, NDQUOT should be more than the maximum number of users that can be logged onto the system. If quotas are in effect, the table entries limit the amount of disk space a user can use. If there are no available entries, the message

```
dquot table full
```

will be printed on the console. If this occurs, the value of `NDQUOT` should be increased.

Profiler

The following parameter is associated with the profiler and is found in the `/etc/master.d/prf` file.

`PRFMAX`	Used by the kernel profiler as a maximum expected number of kernel addresses. This value should be increased if the message `too many text symbols` is printed when `/usr/sbin/prfld` is run.

Interprocess Communication

Messages

The following tunable parameters are associated with interprocess communication messages. These parameters are defined in the `/etc/master.d/msg` file. The order in which they are described follows the order in which they are defined in the output of the `/usr/sbin/sysdef` command.

`MSGMAP`	Specifies the size of the control map used to manage message segments. Default value is 100. Each entry contains 8 bytes.
`MSGMAX`	Specifies the maximum size of a message. The default value is 2048. The maximum size is 64 kilobytes −1.
`MSGMNB`	Specifies the maximum length of a message queue. The default value is 4096.
`MSGMNI`	Specifies the maximum number of message queues systemwide (id structure). The default value is 50.

`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 kilobytes).
`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.
`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 kilobytes).

Semaphores

The following tunable parameters are associated with interprocess communication semaphores. These parameters are defined in the `/etc/master.d/sem` file. The order in which they are described follows the order in which they are defined in the output of the `/usr/sbin/sysdef` command.

`NBPW`	The number of bytes per word. This should not be changed since it is used to calculate the sizes of some tunables.
`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 34 bytes.
`SEMMNS`	Specifies the number of semaphores in the system. The default value is 60. Each entry contains 12 bytes.
`SEMMNU`	Specifies the number of undo structures in the system. The default value is 30. The size is equal to 8 x (`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(2) 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 32,767. The default value 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(2), unless the program has set its own value. The default value is 16,384. The default value is the maximum value for this parameter.

Shared Memory

The following tunable parameters are associated with interprocess communication shared memory. These parameters are defined in the /etc/master.d/shm file. The order in which they are described follows the order in which they are defined in the output of the /usr/sbin/sysdef command.

SHMMAX	Specifies the maximum shared memory segment size. The default value is 131,072.
SHMMIN	Specifies the minimum shared memory segment size. The default value is 1.
SHMMNI	Specifies the maximum number of shared memory identifiers system wide. The default value is 100. Each entry contains 112 bytes.
SHMSEG	There is no maximum value enforced for this tunable (it was 15 in the past). The maximum number of shared memory segments that can be attached per process is dependent on the available unused space the process has.

So even if a process has fewer than SHMSEG shared memory segments, it may not be able to attach another because of its limited space.

Quick Reference to Performance Management

- To collect kernel profiling data:

  `prfdc`

- To collect kernel profiling data at the time of invocation only:

  `prfsnap`

- To collect system activity data automatically:

  `sadc`

- To collect system activity data on demand:

  `sar`

  The following summarizes `sar` options and their results:

□ -a	checks file access operations
□ -b	checks buffer activity
□ -c	checks system calls
□ -d	checks disk activity
□ -g	checks page-out and memory freeing activity
□ -k	checks kernel memory allocation
□ -m	checks interprocess communication
□ -p	checks page-in and fault rates
□ -q	checks queue activity
□ -r	checks unused memory
□ -u	checks CPU utilization
□ -v	checks system table status
□ -w	checks swapping and switching volume
□ -y	checks terminal activity

- `-A` reports overall system performance

- To compress an entire file system (only on a 3B2):

```
/usr/sbin/cmpress
```

- To compress an entire file system:

  `/usr/sbin/dcopy` *fs1 fs2*

- To compress a single directory:

```
mkdir /var/omail
mv /var/mail /var/omail
chmod 777 /var/omail
cd /var/omail
find . -print | cpio -plm ../mail
cd ..
rm -rf omail
```

- To determine the elapsed time, user time, and system time spent in execution of a command:

```
timex
```

- To enable, disable, or check the status of the sampling mechanism:

```
prfstat
```

- To find large, inefficient directories:

```
find / -type d -size +10 -print
```

NOTE `find` thinks in terms of 512-byte blocks.

- To find inactive files:

  `find / -mtime +90 -atime +90 -print >` *filename*

- To format the data collected by `prfdc` or `prfsnap`:

  ```
  prfpr
  ```

- To initialize the kernel-recording mechanism:

  ```
  prfld
  ```

- To move user directories:

  ```
  cd /fs1
  find userx usery -print | cpio -pdm /fs2
  rm -rf /fs1/userx /fs1/usery
  ```

- To print out the number of free file blocks and inodes:

  ```
  df
  ```

- To report disk access location and seek distance:

  ```
  sadp
  ```

- To terminate a runaway process:

  ```
  kill -9
  ```

- To summarize file system usage:

  ```
  du
  ```

- To update the timestamp on the `/stand/system` file:

  ```
  touch/stand/system
  ```

- To use a shell script to collect and store data in the binary file `/var/adm/sa/sa`*dd*:

  ```
  sa1
  ```

- To use a shell script to collect and store data in the ASCII file `/var/adm/sa/sar`*dd*:

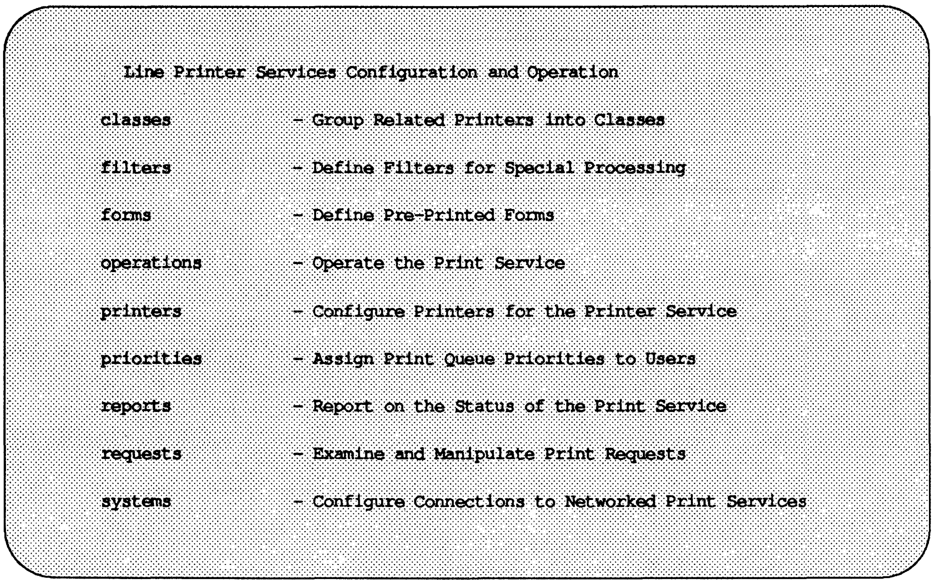
  ```
  sa2
  ```

9 Print Service

Introduction

This chapter describes the administrative tasks required for setting up and running an LP print service. The UNIX system offers a set of menus that help you do administrative tasks such as these. To invoke the "system administration menu" for the LP print service, type `sysadm` and select the `printers` item from the main menu. The following menu will appear on your screen:

Figure 9-1: Main Menu for Print Service



```
   Line Printer Services Configuration and Operation

 classes            - Group Related Printers into Classes

 filters            - Define Filters for Special Processing

 forms              - Define Pre-Printed Forms

 operations         - Operate the Print Service

 printers           - Configure Printers for the Printer Service

 priorities         - Assign Print Queue Priorities to Users

 reports            - Report on the Status of the Print Service

 requests           - Examine and Manipulate Print Requests

 systems            - Configure Connections to Networked Print Services
```

If you prefer not to use the menus, you can perform the same administrative tasks by issuing commands directly to the shell. The following table shows which shell commands are available for doing the tasks listed on the menu.

Figure 9-2: Shell Commands for Print Service Administration

Task Description	Menu Item	Shell Command
Group printers into classes	`classes`	`lpadmin`(1M)
Provide pre-processing software for files to be printed	`filters`	`lpfilter`(1M)
Define pre-printed forms for print requests	`forms`	`lpforms`(1M)
Control (turn on/off) queuing of requests; enable & disable printers; mount forms and fonts; start & stop print service; and report status of printers, classes, & forms	`operations`	`accept` & `reject` [see `accept`(1M)], `enable` & `disable` [see `enable`(1)], `lpadmin`(1M), `lpsched` & `lpshut` [see `lpsched`(1M)], `lpstat`(1)
Configure printers for print service	`printers`	`lpadmin`(1M)
Define levels of priority available to users requesting print jobs	`priorities`	`lpusers`(1M)
Identify active printers, print wheels & character sets, mounted forms, and pending requests	`reports`	`lpstat`(1)
Submit and cancel print requests	`requests`	`lp` & `cancel` [see `lp`(1)], `lpmove` [see `lpsched`(1M)]
Set up communication to remote print service	`systems`	`lpsystem`(1M)

The rest of this chapter describes the work required to set up and maintain print services on a UNIX system with the LP print service utilities. Details about the commands listed above are available in the manual pages for them. Error messages issued by the LP print service are listed in Appendix E.

The information presented in this chapter includes the following:

- A description of how the LP print service works
- References to documentation for installing the print service
- Troubleshooting guidelines
- Instructions for stopping and starting the print service manually
- Instructions for configuring a print service for the unique requirements of your users (such as the need for particular pre-printed forms and filters)
- A list of directories and files delivered as part of the print service
- Instructions for supporting PostScript printers
- Instructions for writing customized filters and interface programs

Overview

The LP print service, originally called the LP spooler, is a set of software utilities that allows you, minimally, to send a file to be printed while you continue with other work. (The term "spool" is an acronym for "simultaneous peripheral output on-line," and "LP" originally stood for Line Printer, but has come to include many other types of printing devices.) The print service has many optional enhancements, however; you can make your print service as simple or as sophisticated as you like.

Components of a Print Service

A print service consists of both hardware and software. You must have at least one computer and one printing device for an LP print service. Beyond that, you may have any number of computers and printing devices; there is no limit to the number of pieces of hardware you may include. The software consists of the LP print service utilities and any filters (programs that process the data in a file before it is printed) that you may provide. Users of your print service may be required to print all their files in the same format, or, if you make different types of printers and/or filters available with your service, they may choose from several formats. You may also offer your users a choice between plain paper and pre-printed forms (such as invoices or checks).

Functions Performed by the Print Service Software

Whether your print service is simple (such as a one-computer/one-printer configuration that prints every file in the same format on the same type of paper) or a sophisticated one (such as a computer network with multiple printers and a choice of printing formats and forms), the LP software helps you maintain it by performing several important functions:

- Scheduling the print requests of multiple users
- Scheduling the work of multiple printers
- Starting programs that interface with the printers
- Filtering users' files (if necessary) so they will be printed properly

- Keeping track of the status of jobs
- Keeping track of forms and print wheels currently mounted and alerting you to mount needed forms and print wheels
- Alerting you to printer problems

Suggestions for LP Print Service Administration

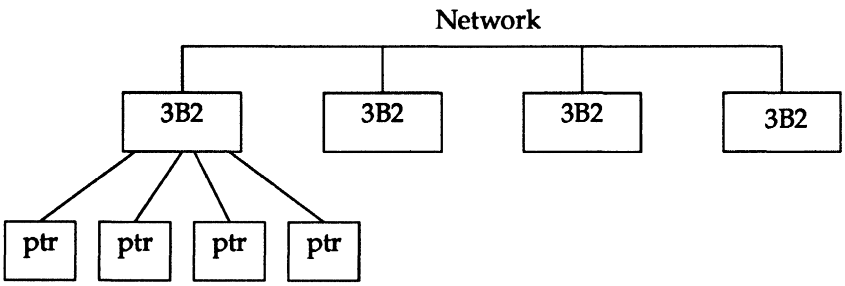
Here are some tips about how to organize your duties as the administrator of an LP print service.

Configuring Your Printer Sites

Where you decide to put your printers and how you decide to connect them to your computers depends on how those printers will be used. (See "Making Printers Accessible Through Your Computer" below for a more detailed description of connection methods.) There are three possible scenarios: (1) users may access printers attached to their own computer; (2) users may access printers attached to a server computer; and (3) users may access remote printers on a network to which their computer belongs.

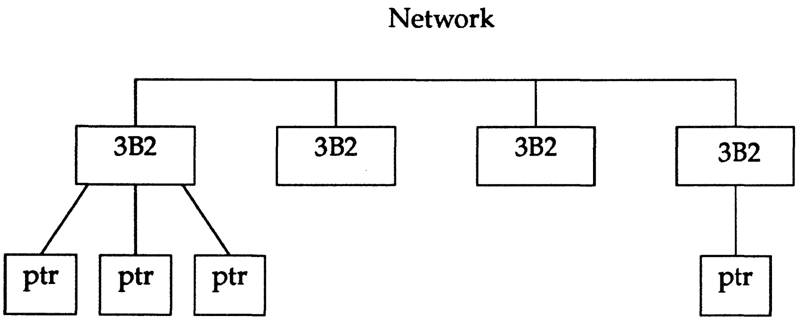
- You may want to connect a particular printer directly to the computer that is the home machine of the users who will use that printer most often. If you do, the type of connection you have will be referred to as a direct connection. An environment that includes more than one computer, each of which has a direct connection to a printer, is said to have a "distributed printing configuration."
- You may want to have all your printers in one physical location, such as a computer center. If so, you might connect them all to one computer. Users on other computers who want to use a printer may access it through a network linking their own computers to the computer serving the printers. An environment in which one computer serves several printers (which can be accessed only through a computer-to-computer network) is described as a "print server configuration." Figure 9-3 shows a sample print server configuration.

Figure 9-3: Print Server Configuration



- You may want to link most of your printers to a dedicated printer server computer, while allowing other printers to be connected to your machine. If so, you can arrange your computers and printers in a network configuration, as shown in Figure 9-4.

Figure 9-4: Network Configuration



Getting Started

Your first tasks are to physically connect your printers to your computers and to install the LP print service utilities from the floppy diskettes on which they were delivered. Once installed, these utilities will be available whenever your UNIX system is brought up. (If for some reason–such as to make a repair–you need to stop the print service without stopping the UNIX system, you will need to stop it manually, and then restart it manually. Instructions for manually stopping and starting the LP print service are provided later in this chapter.)

Even after you've installed the hardware and software, however, printers will not be available for use immediately. Before users can start submitting requests for print jobs, you must complete the following three steps:

- You must configure your printers; that is, you must name the printers and describe their characteristics to the print service. To configure your printers, run the `lpadmin` command. (See "Configuring Your Printers" below for details.)
- You must "register" (with the print service) any printer (or class of printers) that you have configured so that it accepts and queues print requests. To register printers (or classes) run the `accept` command. (See "Making Printers Available" below for details.)
- You must "enable" your printers (that is, you must make your printers active and available to users) by running the `enable` command. (See "Making Printers Available" below for details.)

Installing the LP Print Service

NOTE Before you can install the LP print service software, you must be logged in as `root`, and the system must be in multi-user or single-user state. (If you are in single-user state, you must run the command `mount /usr` before following this procedure.)

- Connect your printer (and any optional hardware you may have) to your computer, following the instructions in the appropriate documents (see "Documentation for Installing Printers" below).
- Install the LP print service software by executing the following command:

  ```
  pkgadd -d diskette1
  ```

  The `pkgadd` command will prompt you to insert the first of the floppy diskettes labeled "LP Spooling Utilities" into the diskette drive at the appropriate time. You will be prompted through the whole installation process.

Sharing Printers

If you have access to other systems via the Remote File Sharing Utilities (RFS), you may want to share printers with those systems by running the print service over RFS. You can do so by following these instructions.

On the server machine:

1. Set up the LP print service on the server machine as you would on any machine. (The server is the computer that does all the spooling.) Make sure that the printer works and that you are able to print text on it.
2. Share `/var/spool/lp`, `/etc/lp`, and `/var/lp` with all the *client(s)* that will be using this printer.
3. In `/etc/rfs/auth.info/uid.rules`, map the user ID (UID) of `lp` to itself, so the entry in `uid.rules` appears as follows: `map lp`.

On the client machines:

1. Do not run the scheduler on the client machines. On the client machines you need only `lp`, `lpstat`, and other LP print service commands.

2. Mount the resources that were shared by the server on the client's `/var/spool/lp`, `/etc/lp`, and `/var/lp`.

On client machines, the `-c` option of `lp` should be used for any user file not in a shared resource. This will force a copy of the file to be sent to the server machine. The LP print service cannot access local files that are not in a shared resource.

Controlling Access to the Print Service

Any user can send requests to the LP print service, check the status of requests, and cancel requests. In addition, you may want to give your users the ability to disable and enable a printer by authorizing them to use the `enable` and `disable` commands. The advantage in doing so is that a user with this authority can turn off a malfunctioning printer without calling the administrator. On the other hand, it may not be reasonable, in your printing environment, to allow regular users to disable printers.

During the installation process, the `pkgadd` command will "ask" you whether you want to authorize the users on your system to enable and disable the printer.

For further instructions on authorizing and restricting access to the `enable` and `disable` commands, see "Allowing Users to Enable and Disable a Printer" (in the "Making Printers Available" section) later in this chapter.

Documentation for Installing Printers

The following documentation can be of specific help in the area of printer operation:

- The installation manual that was delivered with your printer
- *Expanded Input/Output Capability Manual*
- *AT&T 3B2 Computer Installation Manual for AT&T Printers*: This book explains how to set up and use various AT&T printers; it may also help you in setting up other printers.

If you use an early version of this installation manual for printers, keep in mind that the chapter called "Software Installation" refers to the old version of the LP Spooling Utilities. Disregard the instructions referring to the LP software and use, instead, the instructions in the *System Administrator's Guide.*

- *Dot Matrix Printer Manual*
- *Letter Quality Printer Manual*
- *Character User Interface Programmer's Guide*
- `terminfo`(4) in the *Programmer's Reference Manual*

For details about how to order any of these books, see "Related Documents and Training" in "About This Document" at the front of this book.

Configuring Your Printers

Before the LP print service can start accepting print requests, you will have to describe the characteristics of each printer you have. The following is a list of the attributes most commonly defined:

- printer name (mandatory)
- connection method (mandatory for local printers)
- system name (mandatory for access to remote printers and mandatory for allowing remote access to local printers)
- interface program
- printer type
- content types
- printer port characteristics
- character sets or print wheels
- alerting to mount a print wheel
- forms allowed
- printer fault alerting
- printer fault recovery
- restrictions on user access
- inclusion of banner page in output
- printer description
- default printing attributes
- printer class membership
- system default destination
- mounting a form or print wheel
- removing a printer or class

You need to specify very little of this information to add a new printer to the LP print service. The more information you provide, however, the better the printer will satisfy various users' needs.

The descriptions in the sections below will 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 will also be 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've read all the sections before adding a printer, so that you can do it in fewer steps.

Printer Name

The printer name and the connection method (described next) are the only items you must specify to define a new local printer. To define a new remote printer, you must specify the printer name and the system name. The printer name is used to identify the printer, both by you (when you want to change the printer configuration or manage the printer), and by users who want to use the printer. The name may contain a maximum of fourteen alphanumeric characters and underscores.

You may choose any name you like, but it is good practice to choose a name that is meaningful to the users of the LP print service. For example, `laser` is a good name for a laser printer. If you have several laser printers you may name them `laser1`, `laser2`, and so on.

You don't have to try to fit a lot of descriptive information into the name; there is a better place for this information (see the "Printer Description" section below). You also don't have to make the name precisely identify the type of printer; users who need to use a particular type of printer can specify it by type rather than name (see the "Printer Type" section below).

You will use the printer name every time you want to refer to the printer: when adding other configuration information for the printer, when changing the configuration of the printer, when referring to the status of the printer, and when removing the printer. Thus the first thing you must do to add a printer is identify its name. You will do this as shown below; but don't do it yet because you'll also need to specify the connection method.

```
        lpadmin -p printer-name
```

There are no default names; you must name every printer.

Connection Method

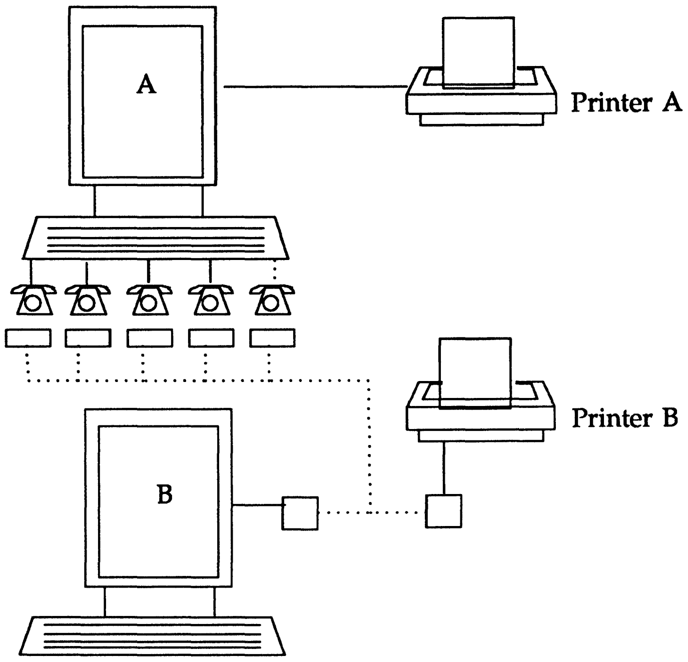
NOTE This section does not apply if you are making a remote printer accessible to users on your system.

The LP print service allows you to connect a printer to your computer in one of the following three ways:

- by connecting the printer directly to your computer
- by connecting the printer directly to a network to which your computer is attached
- by connecting the printer to a modem

Figure 9-5 shows these three types of connections.

Figure 9-5: Methods of Connecting a Printer to a Computer



Computer A accesses printer A through a direct connection, and accesses printer B using modems. Computer B accesses printer B over a local area network. Computer B may be able to access printer A through a remote connection–this is discussed in the next section.

The simplest connection method is by connecting a printer directly to your computer. You may, however, want to connect a printer to a network (so it can be shared with other computers or workstations), or to a modem. Whichever method you use, you must describe it to the LP print service after you've connected the hardware.

To define the connection method for a new printer for your print service, run the `lpadmin` command, specifying a connection method through either the `-v` option for a directly connected printer or the `-U` option for a printer directly connected to a network or a printer connected to a modem.

Direct Connections

The simplest and most common method by which printers are connected to a computer is direct connection. If you use this method, you generally need to specify only two items on the command line when you make the connection: the name of the printer and the name of the connecting port. To connect a printer directly to your computer enter the following command:

```
lpadmin -p printer-name -v pathname
```

where *pathname* is the name of the special device file representing the printer port. The following are examples of typical names of special device files.

```
/dev/contty
/dev/term/11
/dev/term/12
/dev/term/13
/dev/term/14
/dev/term/15
```

(For details about using these files, see "Printer Port Characteristics" later in this chapter.)

Using a Printer As a Login Terminal

Some directly connected printers can also be used as terminals for login sessions. If you want to use a printer as a terminal, you must arrange for the LP print service to handle it as such. To do so, use the `-l` option to the `lpadmin` command, as follows:

```
lpadmin -p printer-name -v pathname -l
```

As before, *pathname* is the name of the special file representing the printer port. If the `-l` option is specified, the printer will be disabled automatically whenever the LP print service is started, and therefore will have to be manually enabled before it can be used for printing. For instructions on manually enabling a printer, see "Enabling and Disabling a Printer" (under "Making Printers Available") later in this chapter.

Connections to Networks and Modems

Why would you want to use a printer that is not directly connected to your computer?

- The environment where a printer is located is so far from the computer that a direct connection is not possible or practical. For example, you might have one printer in use with a single terminal at a branch office located a few miles from your main site.
- You may want to share a printer with computers that are not on a common network.

The LP print service establishes a connection to the printer as necessary to print requests; at the end of each request the connection is dropped, making the printer available to the next machine that calls it. Thus the printer gets shared by the users of all the computers, more or less equally.

There are two methods for connecting printers that are not directly connected to your system: attached directly to a network and through a dial-up modem. The LP print service uses the Basic Networking Utilities (BNU) to handle both methods.

When a modem connection is used, the printer must be connected to a dialed modem, and the dial-out modem must be connected to the computer. Whether printers are connected to a modem or directly to a network, the connection must be described to the Basic Networking Utilities. For instructions on describing either type of connection, see the "Network Services" chapter.

To make a printer connected in one of these ways available to your users, enter the following command:

```
lpadmin -p printer-name -U dial-info
```

Dial-info is either the telephone number to be dialed to reach the printer's modem, or the system name entered in the Basic Networking `Systems` file for the printer.

NOTE The -U option provides a way to link a single printer to your print service. It does not allow you to connect with a print service on another system.

A note on printers connected to a modem or directly to a network: if the printer or port is busy, the LP 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. These rates are 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 the "Fault Alerting" section below), the alert message will give the reason for the fault. These messages are identical to the error messages produced by the Basic Networking Utilities (BNU) for similar problems. See the section titled "BNU STATUS Error Messages" in Appendix E ("Error Messages") for an explanation of the reasons for failure.

In summary, to add printers to your system, run the `lpadmin` command, specifying a connection method through one of two options: the `-v` option for a directly connected printer, or the `-U` option for a networked printer.

System Name

NOTE This section does not apply if you are making only a local printer accessible to users on your system.

A remote printer is one that is connected to a system other than your local system that you can access only via that remote system. Why might you want to use a remote printer? You might have several computers connected via a high-speed network, such as Starlan. You can set up one computer for all the printers.

There are some exceptional cases, however. For example, if only one of the printers has a particular typesetter needed for some print jobs, then users from many systems will want to access it from time to time.

Alternatively, a large community of users on a local area network may want to pool all printers on a single system, where they can share them. When this is done, the system supporting the printers becomes a printer server.

To make accessible a printer that is remote, the name of the system on which the printer resides must be registered with the print service. If the remote printer resides on a System V Release 4 machine, enter

`lpsystem` *system-name*

If the remote printer resides on a BSD machine, enter

`lpsystem -p bsd` *system-name*

NOTE For details about the options available with this command, see `lpsystem`(1M) in the *System Administrator's Reference Manual.*

In either case, after entering the `lpsystem` command, enter the `lpadmin` command, as follows:

`lpadmin -p` *printer* `-s` *system-name*

where *printer* is the name by which your users identify the remote printer and *system-name* is the name of the system on which that printer resides. You can usually use the same name used for that printer by the remote system. If the name used by the remote system is the same name used for an existing printer or class on your system, you must use a different name. To assign a different name to a remote printer, enter the following:

`lpadmin -p` *local-name* `-s` *system-name!remote-name*

For example, imagine you want your users to have access to a printer called `psjet2` that resides on a remote system called `newyork`. Because you already have a printer called `psjet2` on your own system, you want to give the remote printer a new name on your system: `psjet3`. Request the new name by entering the following:

```
lpadmin -p psjet3 -s newyork!psjet2
```

Before you add a remote printer to your system, be sure communications between your system and the network have been set up properly, and verified. See the *Network User's and Administrator's Guide* for details.

Allowing Remote Users to Access Local Printers

Making the printers on your local system accessible to users on remote systems is a two-step process: you must configure the port monitor on the local system and you must register the system with the LP print service. This section provides instructions for these tasks.

Configuring the Local Port Monitor

If the remote system will need access to printers connected directly to your computer, then you need to configure the local port monitor for the network you share to accept service requests and to notify the LP print service of such requests. For System V machines calling your machines, issue the following command:

```
    pmadm -a -p netname -s lp -i root -V 'nlsadmin -V'\
    -m 'nlsadmin -o /var/spool/lp/fifos/listenS5'
```

where *netname* is the name of a network such as `starlan` or `tcp`.

If you expect users on BSD machines to send print requests to your machine, then you need to configure your local port monitor. The output of this command will be a hexadecimal number.

```
    pmadm -a -p tcp -s lpd -i root -V 'nlsadmin -V'\
    -m 'nlsadmin -o /var/spool/lp/fifos/listenBSD -A'\xaddress''
```

Before issuing this command for a BSD machine, however, you need to know its TCP-IP *address*. To get this *address*, run the `-A` option with the `lpsystem` command, as follows:

```
    lpsystem -A
```

Adding a System Entry

If you want your system to accept jobs from a remote system (and vice-versa), the print service must know about that system. The `lpsystem` command allows you to register remote systems with the local print service. Run the command as follows:

```
        lpsystem system-name
```

where *system-name* is the name of the remote system.

Interface Program

NOTE This section does not apply if you are making a remote printer accessible to users on your system.

This is the program the LP print service uses to manage the printer each time a file is printed. It has several 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
- to run a filter that prepares the file for printing
- to manage printer faults

If you do not choose an interface program, the standard one provided with the LP print service will be 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. See "Customizing the Print Service" later in this chapter for details on how to customize an interface program.

If you are using the standard interface program, you needn't specify it when adding a printer. If, however, you will be using a different interface program on a local printer, you can refer to it either by specifying its full pathname or by referring to another printer using the same interface program.

To identify a customized interface program by name, specify the printer name and the pathname of the interface program, as follows:

```
    lpadmin -p printer-name -i pathname
```

To use a customized interface program of another printer, specify the printer names as follows:

```
    lpadmin -p printer-name1 -e printer-name2
```

Printer-name$_1$ is the name of the printer you are adding; *printer-name*$_2$ is the name of an existing printer that is using the customized interface program.

Printer Type

A printer type is the generic name for a printer. Typically it is derived from the manufacturer's name, such as `572` for the AT&T 572 Dot Matrix Printer. When you set up your system you can enhance its ability to serve your users by classifying, on the basis of type, the printers available through the print service. Assigning a "type" for each printer is also important because the LP software extracts information about printers from the `terminfo` database on the basis of type. This information includes the list of the printer's capabilities that is used to check the configuration information you supply to the print service. (By checking information provided by you against the capabilities of the printer, the print service can catch any inappropriate information you may have supplied.) The `terminfo` database also specifies the control data needed to initialize a particular printer before printing a file.

You can assign several types to a printer, if your printer is capable of emulating more than one kind of printer. The AT&T 593 Laser Printer, for instance, can emulate an IBM Proprinter XL, an Epson FX86c, and an HP LaserJetII. The `terminfo` database names these types `593ibm`, `583eps`, and `583hp`, respectively. If you specify more than one printer type, the LP print service will use one of them as appropriate for each print request.

NOTE If you specify more than one printer type, you must specify `simple` as the content type.

While you are not required to specify a printer type, we recommend that you do so; when a printer type is specified, better print services can be provided.

To specify a printer type, use the following command line:

```
        lpadmin -p printer-name -T printer-type-list
```

If you give a list of printer types, separate the names with commas. If you do not define a printer type, the default unknown will be used.

Content Types

While the printer type tells the LP print service what types of printers are being added, the content types tell the LP print service what types of files can be printed directly on each printer. Most printers can print files of two types: the same type as the printer type (if the printer type is defined) and the type `simple`, (meaning an ASCII file) which is the default content type for all printers.

Some printers, though, can accept (and print properly) several different types of files. When adding this kind of printer, specify the names of the content types the new printer accepts by adding these names to the list. (By default, the list contains only one type: `simple`.) If you're adding a remote printer, list the content types that have been established for it by the administrator of the system on which it resides.

To specify the list of content types, enter the following command:

```
        lpadmin -p printer-name -I content-type-list
```

The *content-type-list* is a list of names separated by commas or spaces. If you use spaces to separate the names, enclose the entire list (but not the `-I`) in quotes.

Content type names may look a lot like printer type names, but you are free to choose names that are meaningful to you and the people using the printer. (The names `simple` and any are recognized as having particular meanings by the LP print service; be sure to use them consistently. The name `terminfo` is also reserved, as a reference to *all* types of printers.) The names must contain no more than fourteen characters and may include only letters, digits, and underscores. The following table lists and describes some accepted content types.

Types	Description
`troff`	Device independent output from `troff`
`otroff`	CAT typesetter instructions generated by BSD or pre-System V `troff` (old `troff`)
`tex`	DVI format files
`plot`	Plotting instructions for Tektronix displays and devices
`raster`	Raster bitmap format for Varian raster devices
`cif`	Output of BSD `cifpbt`
`fortran`	ASA carriage control format
`postscript`	PostScript language
`simple`	ASCII file

When a file is submitted to the LP print service for printing with the printer specified by the `-d any` option of the `lp` command, 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 may specify either name (or no name) when submitting a file for printing. If the same content type is printable by several different types of printers, 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.

Most manufacturers produce printers that accept simple ASCII files. While these printers are different types (and thus have different initialization control sequences), they may all be capable of handling the same type of file, which we call `simple`. As another example, several manufacturers may produce printers that accept ANSI X3.64 defined escape sequences. However, the printers may not support all the ANSI capabilities; they may support different sets of capabilities. You may want to differentiate them by assigning different content type names for these printers.

However, while it may be desirable (in situations such as these) to list content types for each printer, it is not always necessary to do so. If you don't, the printer type will be used as the name of the content type the printer can handle. If you have not specified a printer type, the LP print service will assume the printer can print only files of content type `simple`. This may be sufficient if you require users to specify the proper printer explicitly and if files are properly prepared for the printer before being submitted for printing.

The Default Content Type: `simple`

Files of content type `simple` are assumed to contain only two types of characters, printable ASCII characters and the following control characters:

backspace	moves the carriage back one space, except at the beginning of a line
tab	moves the carriage to the next tab stop; by default, stops are spaced every 8 columns on most printers
linefeed	moves the carriage to the beginning of the next line (may require special port settings for some printers—see "Printer Port Characteristics" below)
form feed	moves the carriage to the beginning of the next page
carriage return	moves the carriage to the beginning of the same line (may fail on some printers)

The word "carriage" may be archaic for modern laser printers, but these printers do actions similar to those done by a carriage. If a printer can handle several types of files, including `simple`, you must include `simple` in the content type list; the type `simple` is not automatically added to any list you give. If you *don't* want a printer to accept files of type `simple`, give a blank *content-type-list*, as follows:

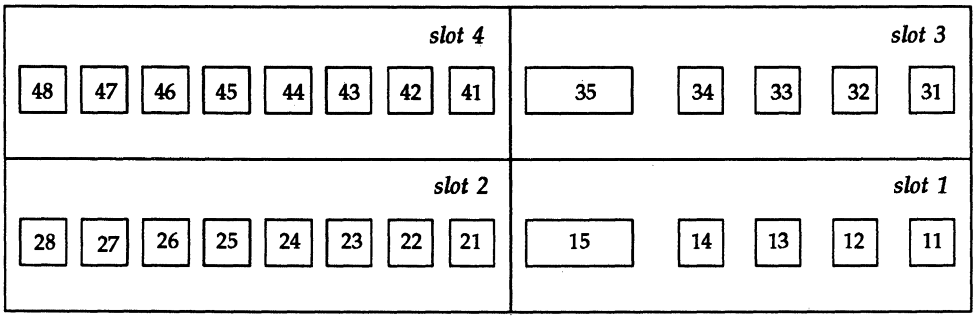
```
        lpadmin -p printer-name -I ""
```

Printer Port Characteristics

NOTE This section does not apply if you are making a remote printer available to users on your system.

For 3B2 Computers

In all 3B2 Computers, the name of a port assignment is made up of two digits: the number of the slot on the computer in which the Expanded I/O card is installed and the number of the port on the card. The following diagram shows a sample of four slots (for Expanded I/O or "ports" cards) as they appear on a 3B2 Computer.



In this example, slots 1 and 3 contain ports cards and slots 2 and 4 contain EPORTS cards. As you can see, ports are numbered from right to left; slots are numbered from right to left, zig-zagging upward from the bottom.

To determine the number of a port, first identify the number of its slot. (Slot 1 will always be in the lower right-hand corner, slot 2 to the left of slot 1, slot 3 above slot 1, and so on.) Use the number of that slot as the first of the two digits of the port assignment number. Next, starting with 1, count the ports on the card in that slot, moving from right to left. Use the number you find in this way as the second digit of the port assignment number.

By counting in this way, you can determine, for example, that the first serial port on the Expanded I/O card in slot number 2 is `/dev/term/21`. Similarly, the parallel port on the Expanded I/O card in slot number 1 is designated `/dev/term/15`. (For more details about ports cards, or "feature cards" as they are also known, see the *AT&T 3B2 Computer Expanded Input/Output Capability Manual*.)

For Any Computer

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; type of parity; and output post-processing. The standard interface program will use 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 below.

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:
`-olcuc`	don't map lower-case to upper-case
`onlcr`	map linefeed into carriage-return/linefeed
`-ocrnl`	don't map carriage-return into linefeed
`-onocr`	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 form-feeds

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 description of the `stty` command in the *User's Reference Manual* 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 will have to customize the interface program. See the section "Customizing the Print Service" for help.

When you add a new printer, you may specify an additional list of port characteristics. The list you give will be applied after the default list, so that you do not need to include in your list items that you don't want to change. Specify the additional list as follows:

```
lpadmin -p printer-name -o "stty='stty-option-list'"
```

Note that both the double quotes and single quotes are needed if you give more than one item in the *stty-option-list*.

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:

```
lpadmin -p printer-name -o "stty=-onlcr"
```

Note that the single quotes are omitted because there's 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:

```
lpadmin -p printer-name -o "stty='parenb parodd cs7'"
```

Character Sets or Print Wheels

NOTE Although your users may use character sets or print wheels that have been mounted on a remote printer (by the administrator of the remote system on which that printer resides), you cannot mount a character set or a print wheel on a remote printer.

Printers differ in the way they can print in different font styles. Some have changeable print wheels, some have changeable font cartridges, others have preprogrammed, selectable character sets.

When adding a printer, you may specify what print wheels, font cartridges, or character sets are available with the printer.

NOTE If you're adding a remote printer and you want your users to be able to use character sets or print wheels that have been mounted by the administrator of the remote system, you must list those character sets and print wheels, just as you would list the character sets and print wheels on a local printer. (See instructions below.)

Only one of these is assumed to apply to each printer. From the point of view of the LP print service, however, print wheels and changeable font cartridges are the same because they require you to intervene and mount a new print wheel or font cartridge. Thus, for ease of discussion, only print wheels and character sets will be mentioned.

When you list the print wheels or character sets available, you will be 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 user to submit a file for printing and ask for a particular font style, without regard for which printer will be 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. Use the following command to determine the names of the character sets listed in the `terminfo` database.

```
tput -T printer-type 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) will 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 LP print service to be able to find the names in the `terminfo` database, you must specify a printer type. See the section "Printer Type" above.

To specify a list of print wheel names when adding a printer, enter the following command.

```
lpadmin -p printer-name -S print-wheel-list
```

The *print-wheel-list* is a comma or space separated list of names. 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:

```
lpadmin -p printer-name -S character-set-list
```

The *character-set-list* is also a comma or space separated list; however, each item in the list looks like one of the following:

`cs`*N*=*character-set-name*
character-set-name$_1$=*character-set-name*$_2$

The *N* in the first case is a number from 0 to 63 that identifies the number of the character set in the `terminfo` database. The *character-set-name*$_1$ 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 may use 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 may refer to a character set by number, by `terminfo` name, 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.

```
tput -T 5310 csnm 1
english
tput -T 5310 csnm 2
finnish
```

The words `english` and `finnish`, the output of the commands, are 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 (set #0) and set #1. You enter the following command to define synonyms.

```
lpadmin -p printer-name -S "cs0=american,\
english=british"
```

The following three commands will then produce identical results.

```
lp -S cs1 -d any ...

lp -S english -d any ...

lp -S british -d any ...
```

If you do not list the print wheels or character sets that can be used with a printer, then the LP print service will assume the following: a printer that takes print wheels has only a single, fixed print wheel, and users may not ask for a special print wheel when using the printer; and 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

NOTE This section does not apply if you are making a remote printer available to users on your system.

If you have printers that can take changeable print wheels, and have listed the print wheels allowed on each, then users will be able to submit a print request to use a particular print wheel. Until it is mounted though (see "Mounting a Form or Print Wheel" in this section), a request for a print wheel will stay queued and will not be printed. You could periodically monitor the number of print requests pending for a particular print wheel, but the LP print service provides an easier way: You can ask to be alerted when the number of requests waiting for a print wheel has exceeded a specified threshold.

You can choose one of several ways to receive an alert.

- You can receive an alert via electronic mail. See the description of the `mailx` command in the *User's Reference Manual* for a description of mail on the UNIX system.
- You can receive an alert written to any terminal on which you are logged in. See the description of the `write` command in the *User's Reference Manual*.
- 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 to see whether any print requests haven't printed because the proper print wheel isn't 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 opt to receive only one alert for each print wheel.

To arrange for alerting to the need to mount a print wheel, enter one of the following commands:

```
lpadmin -S print-wheel-name -A mail -Q requests -W minutes
lpadmin -S print-wheel-name -A write -Q requests -W minutes
lpadmin -S print-wheel-name -A 'command' -Q requests -W minutes
```

The first two commands direct the LP 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 LP print service to run the *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 argument *requests* is the number of requests that need to be waiting for the print wheel before the alert is triggered, and the argument *minutes* is the number of minutes between repeated alerts.

If you do not want the print service to issue an alert when a print wheel needs to be mounted, enter the following:

```
lpadmin -S print-wheel-name -A none
```

NOTE If you want mail sent or a message written to another user for an alert, use the third command with the option `-A 'mail` *user–name*' or `-A 'write` *user–name*'. If you do not specify a *login-name*, the mail or message will be sent to your current ID. This may not be your login ID, if you have used the `su` command to change IDs.

When you start receiving repeated alerts, you can direct the LP print service to stop sending you alerts (for the current case only), by executing the following command.

```
lpadmin -S print-wheel-name -A quiet
```

Once the print wheel has been mounted and unmounted again, alerts will start again if too many requests are waiting. Alerts will 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-wheel-name* is `all` in any of the commands above, the alerting condition will apply to all print wheels for which an alert has already been defined.

If you don't define an alert method for a print wheel, you will not receive an alert to mount it. If you do define a method without the `-W` option, you will be alerted once for each occasion.

Forms Allowed

NOTE For information about how to define, mount, and set up alerting to mount a form, see "Providing Forms" later in this chapter.

You can control the use of preprinted forms on any printer, including remote printers. (Although you cannot mount forms on remote printers, your users may use forms on remote printers.) You may want to do this, for instance, if a printer is not well suited for printing on a particular form because of low print quality, or if the form cannot be lined up properly in the printer.

The LP print service will use a list of forms allowed or denied on a printer to warn you against mounting a form that is not allowed on the printer. However, you have the final word on this; the LP print service will not reject the mounting. The LP print service will, however, reject a user's request to print a file on a printer using a form not allowed on that printer. If, however, the printer is a local printer and the requested form is already mounted, the request will be printed on that form.

If you try to allow a form for a printer, but the printer does not have sufficient capabilities to handle the form, the command will be rejected.

The method of listing the forms 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. (See the description of the `crontab` command in the *User's Reference Manual*.) Briefly, the rules are as follows:

- An allow list is a list of forms that are allowed to be used on the printer. A deny list is a list of forms that are not allowed to be used on the printer.
- If the allow list is not empty, only the forms listed are allowed; the deny list is ignored. If the allow list is empty, the forms listed in the deny list are not allowed. If both lists are empty, there are no restrictions on which forms may be used other than those restrictions that apply to a printer of a particular type, such as a PostScript printer for which a license is required.

- Specifying `all` in the allow list allows all forms; specifying `all` in the deny list denies all forms.

You can add names of forms to either list using one of the following commands:

```
    lpadmin -p printer-name -f allow:form-list
    lpadmin -p printer-name -f deny:form-list
```

The *form-list* is a comma or space separated list of names of forms. If you use spaces to separate names, enclose the entire list (including the `allow:` or `deny:` but not the `-f`) in quotes.

The first command shown above adds names to the allow list and removes them from the deny list. The second command adds names to the deny list and removes them from the allow list. To make the use of all forms permissible, specify `allow:all`; to deny permission for all forms, specify `deny:all`.

If you do not use this option, the LP print service will consider that the printer denies the use of all forms. It will, however, allow you to mount any form, thereby making it implicitly available to use. (See "Mounting a Form or Print Wheel" later in this section for more information.)

Printer Fault Alerting

NOTE This section does not apply if you are making a remote printer accessible to users on your system.

The LP print service provides a framework for detecting printer faults and alerting you to them. 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 a printer failure. The range of fault indicators is also broad, ranging from dropping carrier (the signal that indicates that the printer is on line), to sending an XOFF, to sending a message. Only two classes of printer fault indicators are recognized by the LP print service itself: a drop in carrier and an XOFF not followed in reasonable time by an XON. However, you can add filters that recognize any other printer fault indicators, and rely on the LP 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 "Filter Management" section in this chapter. For a description of how a filter should let the LP 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 the description of the `mailx` command in the *User's Reference Manual* for a description of mail on the UNIX system.
- You can receive an alert written to any terminal on which you are logged in. See the description of the `write` command in the *User's Reference Manual*.
- 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 will need a way of finding out about the faults and fixing them; the LP print service will 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 LP print service cannot automatically determine when a fault has been cleared except by trying to print another file. It will assume that a fault has been cleared when it is successfully able to print a file. Until that time, if you have asked for only one alert per fault, you will not receive another alert. If, after you have fixed a fault, but before the LP print service has tried printing another file, the printer faults again, or if your attempt to fix the fault fails, you will not be notified. Receiving repeated alerts per fault, or requiring manual re-enabling of the printer (see the "Fault Recovery" section below), will overcome this problem.

To arrange for alerting to a printer fault, enter one of the following commands:

```
lpadmin -p printer-name -A mail -W minutes
lpadmin -p printer-name -A write -W minutes
lpadmin -p printer-name -A 'command' -W minutes
```

The first two commands direct the LP 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 LP print service to run the *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* argument is the number of minutes between repeated alerts.

If you do not want the LP print service to issue an alert when a fault occurs, enter the following:

```
lpadmin -p printer-name -A none
```

NOTE If you want mail sent or a message written to another user when a printer fault occurs, use the third command with the option `-A 'mail` *login-name*`'` or `-A 'write` *login-name*`'`. If you do not specify a *login-name*, the mail or message will be sent to your current ID. This may not be your login ID, if you have used the `su` command to change IDs.

Once a fault occurs and you start receiving repeated alerts, you can direct the LP print service to stop sending you alerts (for the current fault only), by executing the following command:

```
lpadmin -p printer-name -A quiet
```

NOTE Use the alert type `quiet` only to terminate an active alert; do not specify `quiet` as the alert type for a new printer.

If the *printer-name* is `all` in any of the commands above, the alerting condition will apply to all printers.

If you don't define an alert method, you will receive mail once for each printer fault. If you define a method without the `-W` option, you will be alerted once for each fault.

Printer Fault Recovery

NOTE This section does not apply if you are making a remote printer accessible to users on your system.

When a printer fault has been fixed and the printer is ready for printing again, the LP print service will recover in one of three ways:

- it will continue printing at the top of the page where printing stopped
- it will restart printing at the beginning of the print request that was active when the fault occurred
- it will wait for you to tell the LP 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 LP print service cannot do this. If a proper filter is not being used, you will be notified in an alert if recovery cannot proceed as you want.

To specify the way the LP print service should recover after a fault has been cleared, enter one of the following commands:

```
lpadmin -p printer-name -F continue
lpadmin -p printer-name -F beginning
lpadmin -p printer-name -F wait
```

These commands direct the LP 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 Printer" section in this chapter for information on the `enable` command).

If you do not specify how the LP print service is to resume after a printer fault, it will try 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 has been cleared, printing will be attempted every few minutes until it succeeds. You can force the LP print service to retry immediately by issuing an `enable` command.

User Access Restrictions

You can control which users are allowed to use a particular printer on your system. For instance, if you're designating one printer to handle sensitive information, you don't want all users to be able to use the printer. Another time you might want to do this is when you're authorizing the use of a high quality printer which produces expensive output.

The LP print service will use a list of users allowed or denied access to a printer. The LP print service will reject a user's request to print a file on a printer he or she is not allowed to use.

NOTE If your users have access to remote printers, or if users on other systems have access to printers on your system, make sure that the allow and deny lists for those printers on your computer match the allow and deny lists on the remote system where the remote printers reside. If these two sets of lists don't match, your users may receive conflicting messages (some accepting jobs, and others refusing jobs) when submitting requests to remote printers.

The method of listing the users allowed or denied access to a printer is similar to the method used to list users allowed or denied access to the `cron` and `at` facilities, and the method described above in the "Forms Allowed" section. Briefly, the rules are as follows:

- An allow list is a list of users allowed to use the printer. A deny list is a list of users denied access to the printer.
- If the allow list is not empty, only the users listed are allowed; the deny list is ignored. If the allow list is empty, users listed in the deny list are not allowed. If both lists are empty, there are no restrictions on who may use the printer.
- Specifying `all` in the allow list allows everybody access to the printer; specifying `all` in the deny list denies access to 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:

```
lpadmin -p printer-name -u allow:user-list
lpadmin -p printer-name -u deny:user-list
```

The *user-list* is a comma or space separated list of names of users. If you use spaces to separate the names, enclose the entire list (including the `allow:` or `deny:` but not the -u) in quotes. Each item in the *user-list* may take any of the following forms:

user	*User* on any system
`all`	All users on all systems
local-system`!`*user*	User on *local-system* only
`!`*user*	User on local system only
`all!`*user*	*User* on any system
`all!all`	All users on all systems
system`!all`	All users on *system*
`!all`	All users on local system

The first command shown above 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.

If you do not use this option, the LP print service will assume that everybody may use the printer.

Inclusion of Banner Page in Output

Most users want to have the output of each print request preceded by a banner page. A banner page shows who requested the printing, the request ID for it, and when the output was printed. It also allows for an optional title that the requester can use to better identify a printout. Finally, the banner page greatly eases the task of separating a sequence of print requests so that each may be given to the correct user.

Sometimes a user needs to avoid printing a banner page. The likely occasions are when the printer has forms mounted that should not be wasted, such as payroll checks or accounts payable checks. Printing a banner page under such circumstances may cause problems.

Enter the following command to allow users to request no banner page:

```
        lpadmin -p printer-name -o nobanner
```

If you later change your mind, you can reverse this choice by entering the following command:

```
        lpadmin -p printer-name -o banner
```

If you do not allow a user to skip the banner page, the LP print service will reject all attempts to avoid a banner page when printing on the printer. This is the default action.

Printer Description

An easy way to give users of the LP print service helpful information about a printer is by adding a description of it. This description can contain any message you'd like, including the number of the room where the printer is found, the name of the person to call with printer problems, and so forth.

Users can see the message when they use the `lpstat -D -p` *printer-name* command.

To add a description of a printer, enter the following command.

```
        lpadmin -p printer-name -D 'text'
```

The *text* is the message. You'll need to include the quotes if the message contains blanks or other characters that the shell might interpret if the quotes are left out.

Default Printing Attributes

The attributes of a printing job include the page size, print spacing (character pitch and line pitch), and `stty` options for that job. If a user requests a job to be printed on a particular form, the printing attributes defined for that form will be used for that job. When, however, a user submits a print request without requesting a form, the print service uses one of several sets of default attributes.

- If the user has specified attributes to be used, those attributes will take precedence.
- If the user has not specified attributes, but the administrator has executed the `lpadmin -o` command, the default attributes for that command will take precedence.
- If neither of the above, the attributes defined in the `terminfo` database for the printer being used will take precedence.

The LP print service lets you override the defaults for each printer. Doing so can make it easier to submit print requests by allowing you to designate different printers as having different default page sizes or print spacing. A user can then simply route a file to the appropriate printer to get a desired style of output. For example, you can have one printer dedicated to printing wide (132-column) output, another printing normal (80-column by 66-line) output, and yet another printing letter quality (12 characters per inch, 8 lines per inch) output.

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 characters and lines, or 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:

```
lpadmin -p printer-name -o width=scaled-number
lpadmin -p printer-name -o length=scaled-number
lpadmin -p printer-name -o cpi=scaled-number
lpadmin -p printer-name -o lpi=scaled-number
```

Append the letter `i` to the *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 don't provide defaults when you configure a printer, then the page size and print spacing will be taken from the data for your printer type in the `terminfo` database. (If you do not specify a printer type, then the type will be `unknown`, for which there is an entry in the `terminfo` database.) You can find out what the defaults will be by first defining the printer configuration without providing your own defaults, then using the `lpstat` command to display the printer configuration. The command

```
        lpstat -p printer-name -l
```

will report the default page size and print spacing.

Printer Class Membership

NOTE This section does not apply if you are making a remote printer accessible to users on your system.

It is occasionally convenient to treat a collection of printers as a single class. The benefit is that a user can submit a file for printing by a member of a class, and the LP print service will pick the first printer in the class that it finds free. This allows faster turn-around, as printers are kept as busy as possible.

Classes aren't needed if the only purpose is to allow a user to submit a print request by type of printer. The `lp -T` *content-type* command allows a user to submit a file and specify its type. The first available printer that can handle the type of the file will be used to print it. The LP print service will avoid using a filter, if possible, by choosing a printer that can print the file directly over one that would need it filtered first.

NOTE See the "Providing Filters" section of this chapter for more information about filters.

Classes do have uses, however. One use 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 LP print service always checks for an available printer in the order 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 LP print service route print requests in the order you wanted.

Until you add a printer to a class, it doesn't belong to one. If you want to do so, use the following command:

```
lpadmin -p printer-name -c class-name
```

If the class *class-name* doesn't exist yet, it will be created. If you want to remove a printer from a class without deleting the printer, enter the following command:

```
lpadmin -p printer-name -r class-name
```

The class name may contain a maximum of fourteen alphanumeric characters and underscores.

NOTE Class names and printer names must be unique. Because they are, a user can specify the destination for a print request without having to know whether it's a class of printers or a single printer.

System Default Destination

You can define the printer or class to be 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.

Make a printer or class the default destination by entering the following command:

```
lpadmin -d printer-or-class-name
```

If you later decide that there should be no default destination, enter a null *printer-or-class-name* as in the following command.

```
lpadmin -d
```

If you don't set a default destination, there will be none. Users will have to explicitly name a printer or class in each print request, (unless they specify the `-T` *content-type* option) or will have to set the `LPDEST` shell variable with the name of a destination.

Mounting a Form or Print Wheel

NOTE See "Providing Forms" later in this chapter for information about pre-printed forms.

Before the LP print service can start printing files that need a preprinted form or print wheel, you must physically mount the form or print wheel on a printer, and notify the LP print service that you have mounted it. (It is not necessary for a form to be included on the allow list in order to mount it.) If alerting has been set on the form or print wheel, you will be alerted when enough print requests are queued waiting for it to be mounted. (See "Alerting to Mount a Form" below and "Alerting to Mount a Print Wheel" above.)

When you mount a form you may want to see if it is lined up properly. If an alignment pattern has been defined for the form, you can ask that this be repeatedly printed after you've mounted the form, until you have adjusted the printer so that the alignment is correct.

Mounting a form or print wheel involves first loading it onto the printer and then telling the LP print service that it is mounted. Because it is difficult to do this on a printer that's currently printing, and because the LP print service will continue to print files not needing the form on the printer, you will probably have to disable the printer first. Thus, the proper procedure is to follow these three steps:

1. Disable the printer, using the `disable` command.
2. Mount the new form or print wheel as described immediately after this list.
3. Re-enable the printer, using the `enable` command. (The `disable` and `enable` commands are described in the "Enabling and Disabling a Printer" section of this chapter.)

First, physically load the new form or print wheel into the printer. Then enter the following command to tell the LP print service it has been mounted. (The following command line is split into two lines for readability.)

```
lpadmin -p printer-name -M -S print-wheel-name\
        -f form-name -a -o filebreak
```

Leave out the `-S` *print-wheel-name* if you are mounting just a form, or leave out the `-f` *form-name* `-a -o filebreak` if you are mounting just a print wheel.

If you are mounting a form with an alignment pattern defined for it, you will be 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 has been defined, you won't be asked to press the RETURN key. You can drop the `-a` and `-o filebreak` options if you don't want to bother with the alignment pattern.

The `-o filebreak` option tells the LP 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:

```
lpadmin -p printer-name -M -S none -f none
```

Leave out the option `-S none` if you just want to unmount a form; likewise, leave out the `-f none` option if you just want to unmount a print wheel.

Until you've mounted a form on a printer, only print requests that don't require a form will be printed. Likewise, until you've mounted a print wheel on a printer, only print requests that don't require a particular print wheel will be printed. Print requests that do require a particular form or print wheel will be held in a queue until the form or print wheel is mounted.

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 cancel them (using the `cancel` command).

Removing the last remaining printer of a class automatically removes the class as well. The removal of a class, however, 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 will no longer have a default destination.

To remove a printer or class, enter the following command:

```
        lpadmin -x printer-or-class-name
```

If all you want to do is to remove a printer from a class without deleting that printer, enter the following command:

```
        lpadmin -p printer-name -r class-name
```

Putting It All Together

It is possible to add a new printer by completing a number of separate steps, shown in the commands described above. You may find it easier, however, to enter one or two commands that combine all the necessary arguments. Below are some examples.

Example 1

Add a new printer called `lp1` (of the type 455) on printer port `/dev/term/13`. It should use the standard interface program, with the default page size of 90 columns by 71 lines, and linefeeds should *not* be mapped into carriage return/linefeed pairs. (The following command line is split into two lines for readability.)

```
lpadmin -p lp1 -v /dev/term/13 -T 455 \
     -o "width=90 length=71 stty=-onlcr"
```

Example 2

Add a new printer called `laser` on printer port `/dev/term/41`. It should use a customized interface program, located in the directory `/usr/doceng/laser_intface`. It can handle three file types—`i10`, `i300`, and `impress`—and it may be used only by the users `doceng` and `docpub`. (The following command line is split into two lines for readability.)

```
lpadmin -p laser -v /dev/term/41 \
     -i /usr/doceng/laser_intface \
     -I "i10,i300,impress" \
     -u "allow:doceng,docpub"
```

Example 3

When you added the `lp1` printer in the first example, you did not set the alerting. Do this now: have the LP print service alert you—by writing to the terminal on which you are logged in—every 10 minutes after a fault until you fix the problem.

```
lpadmin -p lp1 -A write -W 10
```

Examining a Printer Configuration

Once you've defined a printer configuration, you probably want to review it to see if it is correct. If after examining the configuration you find you've made a mistake, just reenter the command that applies to the part that's wrong.

Use the `lpstat` command to examine both the configuration and the current status of a printer. The 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. The long form of the command gives a complete configuration listing.

Enter one of the following commands to examine a printer.

```
lpstat -p printer-name
lpstat -p printer-name -l
```

(The second command is the long form.) With either command you should see one of the following lines of output.

```
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 LP print service failed in trying to use the printer (because of the *reason* shown), and that it will try again later.

With the long form of the command, you should also see the following output:

```
Form mounted: form-name
Content types: content-type-list
Printer type: printer-type
Description: comment
Connection: connection-info
Interface: pathname
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
```

Making Printers Available

There are two steps in making a printer ready for use after you've defined the printer configuration. First, you must instruct the LP print service to accept print requests for the new printer. To do this, run the `accept` command. Second, you must activate or enable the new printer. To do this, run the `enable` command. These tasks are separate steps because you may have occasion to want to do one but not the other.

Accepting Print Requests for a New Printer

Telling the LP print service to accept print requests for the new printer is done with the `accept` command. You will read more about this command in a later section, "Managing the Printing Load." For now, all you need to know is that you should enter the following command to let this printer be used.

```
accept printer-or-class-name
```

As you can see, this command is needed to let the LP print service start accepting print requests for a class, too. To prevent the print service from accepting any more requests, execute the following command.

```
reject printer-or-class-name
```

Enabling and Disabling a Printer

Because you may want to make sure, before printing begins, that the correct form is loaded in your printer, the correct print wheel or font cartridge is in place, and the printer is on-line, the LP print service will wait for an explicit signal from you before it starts printing files. Once you have verified that all the necessary components are in place, you can request the beginning of printing by issuing the `enable` command for a particular printer, as follows:

```
enable printer-name
```

If you want to enable several printers simultaneously, list the printers (separating the names with spaces) on the same line as the `enable` command. Don't enclose the list in quotes.

Disabling a printer stops further print requests from being printed. (It does not, however, stop the LP print service from accepting new print requests for the printer.) From time to time you may want to disable a printer. For example, you may want to interrupt a print request, or you may want to change a form or print wheel, in which case you should disable the printer first. Normally, disabling a printer also stops the request that's currently being printed, placing it back in the queue so it can be printed later. You can, however, have the LP print service wait until the current request finishes, or even cancel the request outright.

To disable a printer, enter one of the following commands:

```
disable -r "reason" printer-name
disable -W -r "reason" printer-name
disable -c -r "reason" printer-name
```

The first command disables the printer, stopping the currently printing request and saving it for printing later. The other commands also disable the printer, but the second one makes the LP print service wait for the current request to finish, while the third cancels the current request.

NOTE The `-c` and `-W` options are not valid when the `disable` command is run to stop a remote printer because, when run for a remote printer, `disable` stops the transferring (rather than the actual printing) of print requests.

The *reason* is stored and displayed whenever anyone checks the status of the printer. You can omit it (and the `-r` option) if you don't want to specify a reason.

Several printers can be disabled at once by listing their names in the same line as the `disable` command.

NOTE You can only enable or disable local printers; the loading of forms, print wheels, and cartridges in a remote printer and the enabling of that printer are the responsibility of the administrator of the remote system. You can, however, enable or disable the transfer of print requests to the remote system on which a printer is located. Only individual printers can be enabled and disabled; classes cannot.

Allowing Users to Enable and Disable a Printer

You may want to make the `enable` and `disable` commands available for use by other users. This availability is useful, for instance, if you have a small organization where anyone who spots a problem with the printer should be able to disable it and fix the problem. This is *not* a good idea if you want to keep others from interfering with the proper operation of the print services.

If you want to allow others access to the `enable` and `disable` commands, use a standard UNIX system feature called the "setuid bit." By assigning ownership of these commands to the user `lp` (this should have been done automatically when you installed the software), and by setting the setuid bit, you can make sure that anyone will be allowed to use the `enable` and `disable` commands. Clearing the bit removes this privilege.

To allow everybody to run `enable` and `disable`, enter the following two commands:

```
chown lp /usr/bin/enable /usr/bin/disable
chmod u+s /usr/bin/enable /usr/bin/disable
```

The first command makes the user `lp` the owner of the commands; this step should be redundant, but it is safer to run the command than to skip it. The second command turns on the setuid bit.

To prevent others from running `enable` and `disable`, enter the following command:

```
chmod u-s /usr/bin/enable /usr/bin/disable
```

Troubleshooting

Here are a few suggestions of what to do if you are having difficulty getting a printer to work.

No Output (Nothing Is Printed)

The printer is sitting idle; nothing happens. First, check the documentation that came with the printer to see if there is a self-test feature you can invoke; make sure the printer is working before continuing.

There are three possible explanations when you don't receive any output: (1) the printer might not be connected to the computer; (2) the printer might not be enabled; or (3) the baud rate for the computer and the printer might not be set correctly. The rest of this section describes each of these situations in detail.

Is the Printer Connected to the Computer?

The type of connection between a computer and a printer may vary. For the AT&T 3B2 computer, the *AT&T 3B2 Computer Installation Manual for AT&T Printers* provides detailed instructions for connecting most AT&T printers. Even if you aren't using an AT&T printer, you may still find this manual helpful.

Is the Printer Enabled?

The printer must be "enabled" in two ways: First, the printer must be turned on and ready to receive data from the computer. Second, the LP print service must be ready to use the printer. If you receive error messages when setting up your printer, follow the "fixes" suggested in the messages. When the printer is set up, issue the commands

```
accept printer-name
enable printer-name
```

where *printer-name* is the name you assigned to the printer for the LP print service. Now submit a sample file for printing:

```
lp -d printer-name file-name
```

Is the Baud Rate Correct?

If the baud rate (the rate at which data are transmitted) is not the same for both the computer and the printer, sometimes nothing will be printed (see below).

Illegible Output

The printer tries printing, but the output is not what you expected; it certainly isn't readable. There are five possible explanations for this situation: (1) the baud rate for the printer might not match the baud rate for the computer; (2) the printer might be connected to an EPORTS board; (3) the parity setting of the computer might be incorrect; (4) the tabs might be set incorrectly; or (5) the printer type might not have been set correctly. The rest of this section describes each of these situations in detail.

Is the Baud Rate Correct?

Usually, when the baud rate of the computer doesn't match that of the printer, you'll get some output but it will not look at all like what you submitted for printing. Random characters will appear, with an unusual mixture of special characters and unlikely spacing.

Read the documentation that came with the printer to find out what its baud rate is. It should probably be set at 9600 baud for optimum performance, but that doesn't matter for now. If it isn't set to 9600 baud, you can have the LP print service use the correct baud rate (by default it uses 9600). If the printer is connected via a parallel port, the baud rate is irrelevant.

To set a different baud rate for the LP print service, enter the following command:

```
        lpadmin -p printer-name -o stty=baud-rate
```

Now submit a sample file for printing (explained earlier in this section).

Is Your Printer Connected to an EPORTS Board?

The baud rate setting of your printer may be important if your 3B2 computer is equipped with an EPORTS board. Transmitted characters are stored temporarily by the printer in a buffer. When the buffer is filled to a specified threshold, the printer sends an XOFF signal to the EPORTS board, asking it to stop transmission. No more characters are sent until space is available in the buffer; then an XON signal is sent (from the printer to the EPORTS board) and transmission resumes.

The EPORTS board is more efficient than other types of ports boards; it sends the same number of characters per second as other types but, unlike other types, it transmits characters continuously rather than intermittently. Therefore, the printer may not be able to send the XOFF signal to the EPORTS board quickly enough to stop transmission before it receives more characters than its buffer can hold. The result is a character overrun of the buffer.

Symptoms of this problem that can be found in output include missing characters and overstriking of lines. Some terminals warn you of a character overrun by sounding an alarm bell or flashing a light. You should be aware, however, that not all terminals are equipped to issue such warnings; you may not notice character overruns until your file has been printed.

If you discover these types of errors in your output, your printer may be incapable of handling high baud rate transmissions sent by your EPORTS board. To stop character overruns, reduce the baud rate on your printer and match it in the LP print service, as described above. If the errors persist, continue reducing the baud rate until your file can be printed without errors.

Is the Parity Setting Correct?

Some printers use a "parity bit" to ensure that the data received for printing has not been garbled in transmission. The parity bit can be encoded in several ways; the computer and the printer must agree on which one to use. If they do not agree, some characters either will not be printed or will be replaced by other characters. Generally, though, the output will look approximately correct, with the spacing of "words" typical for your document and many letters in their correct place.

Check the documentation for the printer to see what the printer expects. The LP print service will not set the parity bit by default. You can change this, however, by entering one of the following commands:

```
lpadmin -p printer-name -o stty=oddp
lpadmin -p printer-name -o stty=evenp
lpadmin -p printer-name -o stty=-parity
```

The first command sets odd parity generation, the second sets even parity. The last command sets the default, no parity.

If you are also setting a baud rate other than 9600, you may combine the baud rate setting with the parity settings, as in the sample command below.

```
lpadmin -p printer-name -o "stty='evenp 1200'"
```

Tabs Set Correctly?

If the printer doesn't expect to receive tab characters, the output may contain the complete content of the file, but the text may appear in a chaotic looking format, jammed up against the right margin (see below).

Correct Printer Type?

See "Wrong Character Set or Font" below.

Legible Printing, but Wrong Spacing

The output contains all of the expected text and may be readable, but the text appears in an undesirable format: double spaced, with no left margin, run together, or zig-zagging down the page. These problems can be fixed by adjusting the printer settings (if possible) or by having the LP print service use settings that match those of the printer. The rest of this section provides details about solving each of these types of problems.

Double Spaced

Either the printer's tab settings are wrong or the printer is adding a linefeed after each carriage return. (The LP print service has a carriage return added to each linefeed, so the combination causes two linefeeds.) You can have the LP print service not send tabs or not add a carriage return by using the `stty -tabs` option or the `-onlcr` option, respectively.)

```
lpadmin -p printer-name -o stty=-tabs
lpadmin -p printer-name -o stty=-onlcr
```

No Left Margin/Runs Together/Jammed Up

The printer's tab settings aren't correct; they should be set every 8 spaces. You can have the LP print service not send tabs by using the `-tabs` option.

```
lpadmin -p printer-name -o stty=-tabs
```

Zig Zags Down the Page

The `stty onlcr` option is not set. This is set by default, but you may have cleared it accidentally.

```
lpadmin -p printer-name -o stty=onlcr
```

A Combination of Problems

If you need to use several of these options to take care of multiple problems, you can combine them in one list, as shown in the sample command below. Include any baud rate or parity settings, too.

```
lpadmin -p printer-name -o "stty='-onlcr -tabs 2400'"
```

Correct Printer Type?

See below.

Wrong Character Set or Font

If the wrong printer type was selected when you set up the printer with the LP print service, the wrong "control characters" can be sent to the printer. The results are unpredictable and may cause output to disappear or to be illegible, making it look like the result of one of the problems described above. Another result may be that the wrong control characters cause the printer to set the wrong character set or font.

If you don't know which printer type to specify, try the following to examine the available printer types. First, if you think the printer type has a certain name, try the following command:

```
tput -T printer-type longname
```

(This may not work on early versions of AT&T UNIX System V.) The output of this command will appear on your terminal: a short description of the printer identified by the *printer-type*. Try the names you think might be right until you find one that identifies your printer.

If you don't know what names to try, you can examine the `terminfo` directory to see what names are available. Warning: There are probably many names in that directory. Enter the following command to examine the directory.

```
ls -R /usr/share/lib/terminfo/*
```

Pick names from the list that match one word or number identifying your printer. For example, the name `495` would identify the AT&T 495 Printer. Try each of the names in the other command above.

When you have the name of a printer type you think is correct, set it in the LP print service by entering the following command:

```
lpadmin -p printer-name -T printer-type
```

Dial Out Failures

The LP print service uses the Basic Networking Utilities to handle dial out printers. If a dialing failure occurs and you are receiving printer fault alerts, the LP print service reports the same error reported by the Basic Networking software for similar problems. (If you haven't arranged to receive fault alerts, they are mailed, by default, to the user `lp`.) See the "Basic Networking Utilities Error Messages" section of Appendix E in this guide for an explanation of the failure reasons.

Idle Printers

There are several reasons why you may find a printer idle and enabled but with print requests still queued for it:

- The print requests need to be filtered. Slow filters run one at a time to avoid overloading the system. Until a print request has been filtered (if it needs slow filtering), it will not print. Use the following command to see if the first waiting request is being filtered.

  ```
  lpstat -o -l
  ```

- The printer has a fault. After a fault has been detected, printing resumes automatically, but not immediately. The LP print service waits about five minutes before trying again, and continues trying until a request is printed successfully. You can force a retry immediately by enabling the printer as follows:

  `enable` *printer-name*

- A dial out printer is busy or doesn't answer, or all dial out ports are busy. As with automatic continuation after a fault, the LP print service waits five minutes before trying to reach a dial out printer again. If the dial out printer can't be reached for an hour or two (depending on the reason), the LP print service finally alerts you to a possible problem. You can force a retry immediately by enabling the printer as follows:

  `enable` *printer-name*

Networking Problems

You may encounter several types of problems while trying to get files printed over a network: (1) requests being sent to remote printers may back up in the local queue; (2) requests sent to remote printers may be backed up in the remote queue; or (3) a user may receive contradictory messages about whether a remote printer has accepted a print request. The rest of this section describes each of these situations and suggests how to resolve them.

Jobs Backing Up in the Local Queue

There are a lot of jobs backing up in the local queue for a remote printer. There are three possible explanations:

- The remote system is down or the network between the local and remote systems is down. To resolve this problem, run the `reject` command for all the remote printers on your system, as follows:

  `reject` *printer-name*

  This will stop new requests for those printers from being added to the queue. Once the system comes up again, and jobs start being taken from your queue, type `accept` *printer* to allow new jobs to be queued.
- The remote printer is disabled on the local system.
- The underlying System V network software was not set up properly. For details, see `lpsystem`(1M).

Jobs Backing Up in the Remote Queue

The remote printer has been disabled.

Conflicting Messages About the Acceptance/Rejection of Jobs

A user enters a print request and is notified that the system has accepted it. The job is sent to a remote system and the user receives mail that the job has been rejected. This may be happening for one of two reasons. First, the local computer may be accepting requests while the remote computer is rejecting requests.

Second, the definition of the remote printer on the local computer may not match the definition of that printer on the remote computer. Specifically, the definitions of print job components such as filters, character sets, print wheels, and forms are not the same on the local and remote systems. Identical definitions of these job components must be registered on both the local and the remote systems, if local users are to be able to access remote printers.

Providing Forms

A form is a sheet of paper, on which text or graphical displays have already been printed, that can be loaded into a local printer (that is, a printer on your system) for use in place of plain stock. Common examples of forms include company letterhead, special paper stock, invoices, blank checks, vouchers, receipts, and labels.

Typically, several copies of a blank form are loaded into a printer, either as a tray of single sheets or as a box of fan-folded paper. An application is used to generate data that will be printed on the form, thereby filling it out.

The LP print service helps you manage the use of preprinted forms, but does not provide your application any help in filling out a form; this is solely your application's responsibility. The LP print service, however, will keep track of which print requests need special forms mounted and which forms are currently mounted. It can alert you to the need to mount a new form.

This section tells you how you can manage the use of preprinted forms with the LP print service. You will see how you can

- define a new form
- change the print service's description of an existing form
- remove the print service's description of a form
- examine the print service's description of a form
- restrict user access to a form
- arrange alerting to the need to mount a form
- inform the print service that a form has been mounted

Defining a Form

When you want to provide a new form, the first thing you have to do is define its characteristics. To do so, enter information about each of the nine required characteristics (page length, page width, and so on) as input to the `lpforms` command (see below for details). The LP print service will use this information for two purposes: to initialize the printer so that printing is done properly on the form, and to send you reminders about how to handle that form. Before running the `lpforms` command, gather the following information about your new form:

Page length	The length of the form, or of each page in a multi-page form. This can be expressed as the number of lines, or the size in inches or centimeters.
Page width	The width of the form, expressed in characters, inches, or centimeters.
Number of pages	The number of pages in a multi-page form. The LP print service uses this number with a filter (if available) to restrict the alignment pattern to a length of one form. (See the description of alignment patterns below.) If no filter is available, the LP print service does not truncate the output.
Line pitch	A measurement that shows how closely together separate lines appear on the form. It can be expressed in either lines per inch or lines per centimeter.
Character pitch	A measurement that shows how closely together separate characters appear on the form. It can be expressed in either characters per inch or characters per centimeter.
Character set choice	The character set, print wheel, or font cartridge that should be used when this form is used. A user may choose a different character set for his or her own print request when using this form, or you can insist that only one character set be used.
Ribbon color	If the form should always be printed using a certain color ribbon, then the LP print service can remind you which color to use when you mount the form.
Comment	Any remarks that might help users understand what the form is, when it should be used, and so on.
Alignment pattern	A sample file that the LP print service uses to fill one blank form. When mounting the form, you can print this pattern on the form to align it properly. You can also define a content type for this pattern so that the printer service knows how to print it.

NOTE The LP print service does not try to mask sensitive information in an alignment pattern. If you do not want sensitive information printed on sample forms – very likely the case when you align checks, for instance – then you should mask the appropriate data. The LP print service keeps the alignment pattern stored in a safe place, where only you (that is, the user `lp` and the super-user `root`) can read it.

When you've gathered this information about the form, enter it as input to the `lpforms` command. You may want to record this information first in a separate file so you can edit it before entering it with `lpforms`. You can then use the file as input instead of typing each piece of information separately after a prompt. Whichever method you use, enter the information in the following format:

```
Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice: character-set-name[,mandatory]
Ribbon color: ribbon-color
Comment:
informal notes about the form
Alignment pattern: [content-type]
alignment-pattern
```

Although these attributes are described in detail on the previous page, a few points should be emphasized here. First, the phrase `[mandatory]` is optional and, if present, means that the user cannot override the character set choice in the form. Second, *content-type* can be given optionally, with an alignment pattern. If this attribute is given, the print service uses it to determine, as necessary, how to filter and print the file.

With two exceptions, the information in the above list may appear in any order. The exceptions are the alignment pattern (which must always appear last) and the *comment* (which must always follow the line with the `Comment:` prompt). If the *comment* contains a line beginning with a key phrase (such as `Page length`, `Page width`, and so on), precede that line with a > character so the key phrase is hidden. Be aware, though, that any initial > will be stripped from the comment when it is displayed.

Not all of the information has to be given. When you don't specify values for the items listed below, the values shown beside them are assigned by default.

Item	Default
Page length	66 lines
Page width	80 columns
Number of pages	1
Line pitch	6 per inch
Character pitch	10 per inch
Character set choice	any
Ribbon color	any
Comment	(no default)
Alignment pattern	(no default)

To define the form, use one of the following commands

```
        lpforms -f form-name -F file-name
        lpforms -f form-name -
```

where *file-name* is the full path for the file.

The first command gets the form definition from a file; the second command gets the form definition from you, through the standard input. A *form-name* can be anything you choose, as long as it contains a maximum of fourteen alphanumeric characters and underscores.

If you need to change a form, just reenter one of the above commands. You need only provide information for items that must be changed; items for which you don't specify new information will stay the same.

Removing a Form

The LP print service imposes no fixed limit on the number of forms you may define. It is a good idea, however, to remove forms that are no longer appropriate. If you don't, users will see a long list of obsolete forms when choosing a form, and may be confused. In addition, because the LP print service must occasionally look through all the forms listed before performing certain tasks, the failure to remove obsolete forms may require extra, unnecessary processing by the print service.

To remove a form, enter the following command:

```
        lpforms -f form-name -x
```

Restricting User Access

If your system has a form that you don't want to make available to everyone, you can limit its availability to selected users. For example, you may want to limit access to checks to the people in the payroll department or accounts payable department.

The LP print service restricts the availability of a form by using the lists (provided by you) of users allowed or denied access to that form. If a user is not allowed to use a particular form, the LP print service will reject his or her request to print a file with it.

The method used to allow or deny users access to a form is similar to the method used to allow or deny users access to the `cron` and `at` facilities. (See the description of the `crontab` command in the *User's Reference Manual.*) Briefly, the rules are as follows:

- An allow list is a list of users who are allowed to use the form. A deny list is a list of users who are not allowed to use the form.
- If the allow list is not empty, only the users listed are allowed; the deny list is ignored. If the allow list is empty, the users listed in the deny list are not allowed to to use the form. If both lists are empty, there are no restrictions on who may use the form.
- Specifying `all` in the allow list allows everybody to use the form; specifying `all` in the deny list allows no one except the user `lp` and the super-user (`root`) to use the form.

If users on your system are to be able to access forms on a remote printer, it's necessary for all the users included on the allow list for the local system to be included on the allow list for the remote system, as well.

If, on the other hand, a local user is to be denied permission to use forms on a remote printer, it's not necessary for the deny lists on both the local and remote print services to include that user. By being included in only one of these deny lists, a user can be denied access to remote forms. As a courtesy to your users, however, it's a good idea to make sure that any local users who are included in a deny list on a remote system are included in the corresponding deny list on your local system. By doing this you can make sure that whenever a user on your system requests a form that he or she is not authorized to use, he or she is immediately informed that permission to use the form is being denied. If the local print service does not "know" that a user is denied permission to use a particular remote form, there will be a delay before the user receives a "permission denied" message from the remote system.

You can add names of users to either list, using one of the following commands:

```
lpforms -f form-name -u allow:user-list
lpforms -f form-name -u deny:user-list
```

The *user-list* is a comma or space separated list of names of users. If you use spaces to separate the names, enclose the entire list (including the `allow:` or `deny:` but not the `-u`) in quotes. Each item in the list can include a system name, as shown under "User Access Restrictions" earlier in this chapter. 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. Specifying `allow:all` will allow everybody; specifying `deny:all` will deny everybody.

If you do not add user names to the allow or deny lists, the LP print service will assume that everybody may use the form.

Alerting to Mount a Form

If you define more forms than printers, you will obviously not be able to print files on all the forms simultaneously. This means that some print requests may be held in a queue until you mount the forms they need. How will you know when to mount a particular form? One method would be to monitor, periodically, the number of print requests pending for that form. The LP print service, however, provides an easier way: You can ask to be alerted when the number of requests waiting for a form has exceeded a specified threshold.

You can choose one of several ways to receive an alert.

- You can receive an alert via electronic mail. (See the description of the `mailx` command in the *User's Reference Manual* for a description of mail on the UNIX system.)
- You can receive an alert written to any terminal on which you are logged in. (See the description of the `write` command in the *User's Reference Manual*.)
- 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 to see whether any print requests haven't been printed because the proper form isn't 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 form is mounted. You can choose the rate of repeated alerts, or choose to receive only one alert for each form.

To arrange for alerting to the need to mount a form, enter one of the following commands:

```
lpforms -f form-name -A mail -Q requests -W minutes
lpforms -f form-name -A write -Q requests -W minutes
lpforms -f form-name -A 'command' -Q requests -W minutes
```

The first two commands direct the LP 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 LP print service to run the *command* for each alert. The shell environment 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 the current directory.

In each command line, the argument *requests* is the number of requests that need to be waiting for the form to trigger the alert, and the argument *minutes* is the number of minutes between repeated alerts.

NOTE If you want mail sent or a message written to another user for an alert, use the third command with the option `-A 'mail` *login-name*`'` or `-A 'write` *login-name*`'`. If you do not specify a login-name, the mail or message will be sent to your current ID. This may not be your login ID, if you have used the `su` command to change IDs.

If you want the print service to issue no alert when the form needs to be mounted, execute the following command:

```
lpforms -f form-name -A none
```

When you start receiving repeated alerts, you can direct the LP print service to stop sending you alerts (for the current case only) by issuing the following command:

```
lpforms -f form-name -A quiet
```

NOTE Use the alert type `quiet` only to terminate an active alert; do not specify `quiet` as the alert type for a new printer.

Once the form has been mounted and unmounted again, alerts will resume if too many requests are waiting. Alerts will also start again if the number of requests waiting falls below the `-Q` threshold and then rises up to the `-Q` threshold again. This happens when waiting requests are canceled, and when the type of alerting is changed.

If *form-name* is `all` in any of the commands above, the alerting condition applies to all forms for which an alert has not already been defined.

If you don't define an alert method for a form, you will not receive an alert to mount it. If you define a method without the `-W` option, you will be alerted once for each occasion.

Mounting a Form

Refer to the section "Mounting a Form or Print Wheel" under "Configuring Your Printers" in this chapter.

Examining a Form

Once you've defined a form to the LP print service, you can examine it with one of two commands, depending on the type of information you want to check. The `lpforms` command displays the attributes of the form. (The display produced by `lpforms` can be used as input; you may want to save it in a file for future reference.) The `lpstat` command displays the current status of the form.

Enter one of the following commands to examine a defined form.

```
lpforms -f form-name -l lpforms -f form-name -l > file-name
lpstat -f form-name lpstat -f form-name -l
```

The first two commands present the definition of the form; the second command captures this definition in a file, which can be used later to redefine the form if you inadvertently remove the form from the LP print service. The last two commands present the status of the form, with the second of the two giving a long form of output, similar to the output of `lpforms -l`:

```
Page length: scaled-number Page width: scaled-number Number of
pages: integer Line pitch: scaled-number Character pitch:
scaled-number Character set choice: character-set[,mandatory]
Ribbon color: ribbon-color Comment: comment Alignment pattern:
[content-type] content
```

To protect potentially sensitive content, the alignment pattern is not shown if the `lpstat` command is used.

Providing Filters

This section explains how you can manage the use of filters with the LP print service. You will see how you can

- define a new filter
- change a filter
- remove a filter
- examine a filter

The "Customizing the Print Service" section at the end of this chapter describes how you can write a filter. First, let's see what a filter is and how the LP print service can use one.

What Is a Filter?

A filter is a program that you can use for any of three purposes:

- To convert a user's file from one data format to another so that it can be printed properly on a given printer
- To handle the special modes of printing that users may request with the `-y` option to the `lp` command (such as two-sided printing, landscape printing, draft or letter quality printing)
- To detect printer faults and notify the LP print service of them, so that the print service can alert you

Not every filter can perform all three tasks. Given the printer-specific nature of these three roles, the LP print service has been designed so that these roles can be implemented separately. This separation allows you or a printer manufacturer (or another source) to provide filters without having to change the LP print service.

A default filter is provided with the LP print service to provide simple printer fault detection; it does not convert files or handle any of the special modes. It may, however, be adequate for your needs.

Let's examine the three tasks performed by filters more closely.

Task 1: Converting Files

For each printer (local or remote) you can specify what file content types it can print. When a user submits a file to print on any printer, and specifies its content type, the print service will find a printer that can handle files of that content type. Because many applications can generate files for various printers, this is often sufficient. However, some applications may generate files that cannot be printed on your printers.

By defining and creating a filter that converts such files into a type that your printers can handle, you can begin to support more applications in the LP print service. (The LP print service comes with a few filters for converting various types of files into PostScript.) For each filter you add to the system, you must specify one or more types of input it can accept and the type of output it can produce (usually only one).

When a user specifies (by executing `lp -T`) a file content type that no printer can handle, the print service tries to find a filter that can convert the file into an acceptable type. If the file to be printed is passed through a filter, the print service will then match the output type of that filter with a printer type or the input type of another filter. The LP print service will continue to match output types to input types in this way, thus passing a file through a series of filters, until the file reaches a printer that accepts it.

Below are some examples.

Example 1

The user Chris has run a spreadsheet program and has generated a file containing a copy of a spreadsheet. Chris now wants to print this file using the LP print service. You have only AT&T model 455 printers on your system. Fortunately, the spreadsheet application understands how to generate output for several printers, and Chris knows it's necessary to request output that can be handled by the AT&T 455. When Chris submits the file for printing, the LP print service queues it for one of the printers; no filter is needed.

Example 2

A user named Marty has created a graphic image that can be displayed on a Tektronix 4014 terminal. Marty now wants to print this image, but all of the printers are PostScript printers. Fortunately, your system provides a filter called `posttek` that converts Tektronix type files to PostScript. Because you set the printer type to PostScript, the LP print service recognizes that it can use the `posttek` filter to convert Marty's output before printing it.

Task 2: Handling Special Modes

Another important role that filters can perform is the handling of special printing modes. Each filter you add to the filter table can be registered to handle special modes and other aspects of printing:

special modes
printer type
character pitch
line pitch
page length
page width
pages to print
character set
form name
number of copies

A filter is required to handle the special modes and printing of specific pages; the LP print service provides a default handling for all the rest. However, it may be more efficient to have a filter handle some of the rest, or it may be that a filter has to know several of these aspects to fulfill its other roles properly. A filter may need to know, for example, the page size and the print spacing if it is going to break up the pages in a file to fit on printed pages. As another example, some printers can handle multiple copies more efficiently than the LP print service, so a filter that can control the printer can use the information about the number of copies to skip the LP print service's default handling of multiple copies.

We'll see below how you can register special printing modes and other aspects of printing with each filter.

Task 3: Detecting Printer Faults

Just as converting a file and handling special printing modes is a printer-specific role, so is the detecting of printer faults. The LP print service attempts to detect faults in general, and for most printers it can do so properly. The range of faults that the print service can detect by itself, however, is limited. It can check for hang-ups (loss of carrier, the signal that indicates the printer is on-line) and excessive delays in printing (receipt of an XOFF flow-control character to shut off the data flow, with no matching XON to turn the flow back on). However, the print service can't determine the cause of a fault, so it can't tell you what to look for.

A properly designed filter can provide better fault coverage. Some printers are able to send a message to the host describing the reason for a fault. Others indicate a fault by using signals other than the dropping of a carrier or the shutting off of data flow. A filter can serve you by detecting more faults and providing more information about them than you would otherwise receive.

Another service a filter can provide is to wait for a printer fault to clear and then to resume printing. This service allows for more efficient printing when a fault occurs because the print request that was interrupted does not have to be reprinted in its entirety. Only a real filter, which has knowledge of the control sequences used by a printer, can "know" where a file breaks into pages; thus only such a filter can find the place in the file where printing should resume.

The LP print service has a simple interface that allows a filter to send you fault information and to restart printing if it can. The alerting mechanism (see the "Printer Fault Alerting" section under "Configuring Your Printers" in this chapter) is handled by the LP print service; the interface program that manages the filter takes all error messages from the filter and places them in an alert message that can be sent to you. Thus you'll see any fault descriptions generated by the filter. If you've set the printer configuration so that printing should automatically resume after a fault is cleared, the interface program will keep the filter active, so that printing can pick up where it left off.

Will Any Program Make a Good Filter?

It is tempting to use a program such as `troff`, `nroff`, or a similar word-processing program as a filter. However, the `troff` and `nroff` programs have a feature that allows references to be made in a source file to other files, known as "include files." The LP print service does not recognize include files; it will not enqueue any that are referenced by a source file when that file is in a queue to be printed. As a result, the `troff` or `nroff` program, unable to access the include files, may fail. Other programs may have similar features that limit their use as filters.

Here are a few guidelines for evaluating a program for use as a filter:

- Only programs capable of reading data from standard input and writing data to standard output may be used as filters.
- Examine the kinds of files users will submit for printing that will require processing by the program. If they stand alone (that is, if they do not reference other files that the program will need), the program is probably okay.

  Check also to see if the program expects any files other than those submitted by a user for printing. If it does, those files must be in the directory of the person using the filter, or they must be readable by all users authorized to use the filter. The latter prerequisite is necessary because filters are run with the user ID and group ID of the user who submitted the print request.
- If referenced files are permitted in the files submitted for printing, or if the program will need files other than those submitted by a user, then the program, unable to access the additional files, is likely to fail. We suggest you don't use the program under consideration as a filter; instead, have users run the program before submitting files for printing.

Referenced files that are always specified by full pathnames *may* be okay, but only if the filter is used for local print requests. When used on requests submitted from a remote machine for printing on your machine, the filter may still fail if the referenced files exist only on the remote machine.

Filters for Your System

The LP print service is delivered with several filters. As you add, change, or delete filters, you may overwrite or remove some of these original filters. If necessary, you can restore the original set of filters (and remove any filters you have added), by running the following command:

```
lpfilter -f all -i
```

Defining a Filter

When adding a new filter, the first thing you must do is to define the characteristics of its use. To do this, issue the `lpfilter` command with arguments that specify the values of the following filter characteristics:

- the name of the filter (that is, a command name)
- the types of input it will accept
- the types of output it will produce
- the types of printers to which it will be able to send jobs
- the names of specific printers to which it will send jobs
- the "type" of the filter (whether it's a `fast` filter or a `slow` filter)
- options

Each of these characteristics is described below.

`Command:`	This is the full path of the filter program. If there are any fixed options that the program always needs, include them here.
`Input types:`	This is the list of file content types that the filter can process. The LP print service doesn't impose a limit on the number of input types that can be accepted by a filter, but most filters can take only one. Several file types may be similar enough so that the filter can deal with them. You can use whatever names you like here, using a maximum of fourteen alphanumeric characters and dashes (not underscores). Because the LP print service uses these names to match a filter with a file type, you should follow a consistent naming convention. For example, if more than one filter can accept the same input type, use the same name for that input type when you specify it for each filter. These names should be advertised to your users so they know how to identify the type of a file when submitting that file for printing.
`Output types:`	This is the list of file types that the filter can produce as output. For each input type the filter will produce a single output type, of course; the output type may vary, however, from job to job. The names of the output types are also restricted to fourteen alphanumeric characters and dashes.

These names should either match the types of printers you have on your system, or match the input types handled by other filters. The LP print service groups filters together in a shell pipeline if it finds that several passes by different filters are needed to convert a file. It's unlikely that you will need this level of sophistication, but the LP print service allows it. Try to find a set of filters that takes (as input types) all the different files your users may want printed, and converts those files directly into types your printers can handle.

`Printer types:`

This is a list of printer types into which the filter can convert files. For most filters this list will be identical to the list of output types, but it can be different.

For example, you may have a printer that is given a single type for purposes of initialization (see the "Printer Type" section under "Printer Management" in this chapter), but which can recognize several different types of files. In essence this printer has an internal filter that converts the various types into one with which it can deal. Thus, a filter may produce one of several output types that match the "file types" that the printer can handle. The filter should be marked as working with that printer type.

As another example, you may have two different models of printers that are listed as accepting the same types of files. However, due to slight differences in manufacture, one printer deviates in the results it produces. You label the printers as being of different printer types, say A and B, where B is the one that deviates. You create a filter that adjusts files to account for the deviation produced by printers of type B. Because this filter is needed only for those printer types, you would list it as working only on type B printers.

For most printers and filters you can leave this part of the filter definition blank.

`Printers:` You may have some printers that, although they're of the correct type for a filter, are in other ways not adequate for the output that the filter will produce. For instance, you may want to dedicate one printer for fast turn-around; only files that the printer can handle without filtering will be sent to that printer. Other printers, of identical type, you allow to be used for files that may need extensive filtering before they can be printed. In this case, you would label the filter as working with only the latter group of printers.

In most cases a filter should be able to work with all printers that accept its output, so you can usually skip this part of the filter definition.

`Filter type:` The LP print service recognizes "fast" filters and "slow" filters. Fast filters are labeled "fast" because they incur little overhead in preparing a file for printing, and because they must have access to the printer when they run. A filter that is to detect printer faults has to be a fast filter. A filter that uses the `PRINTER` keyword as a filter option must be installed as a fast filter.

Slow filters are filters that incur a lot of overhead in preparing a file and that don't require access to a printer. The LP print service runs slow filters in the background, without tying up a printer. This allows files that don't need slow filtering to move ahead; printers will not be left idle while a slow filter works on a file if other files can be printed simultaneously.

Slow filters that are invoked by modes (via the `-y` option), must be run on the computer where the print request was issued. The LP print service can't pass values for modes to remote machines. It can, however, match a file content type (specified after the `-T` option of the `lp` command) to a content type on a remote machine. Therefore, if you want to activate special modes on a remote machine, you must do so by specifying content types that will allow the LP print service to match input types and output types.

`Options:` Options specify how different types of information should be transformed into command line arguments to the filter command. This information may include specifications from a user (with the print request), the printer definition, and the specifications implemented by any filters used to process the request.

There are thirteen sources of information, each of which is represented by a "keyword." Each option is defined in a "template," a statement in the following format:

keyword pattern=replacement.

This type of statement is interpreted by the LP print service to mean "When the information referred to by *keyword* has the value matched by *pattern,* take the *replacement* string, replace any asterisks it contains with the *pattern* specified or expand any regular expressions it contains, and append the result to the command line."

The options specified in a filter definition may include none, all, or any subset of these thirteen keywords. In addition, a single keyword may be defined more than once, if multiple definitions are required for a complete filter definition. (See "Defining Options with Templates" below.)

When you've gathered enough information to define the above characteristics of your filter, you are ready to run the `lpfilter` command, using your data as arguments. Because there are so many arguments, and because some of them may need to be entered more than once (with different values), we recommend you record this information first in a separate file and edit it, if necessary. You can then use the file as input to the `lpfilter` command and avoid typing each piece of information separately.

Whether you store the information in a file or enter it directly on the command line, use the following format:

`Command:` *command-pathname* [*options*] `Input types:` *input-type-list* `Output types:` *output-type-list* `Printer types:` *printer-type-list* `Printers:` *printer-list* `Filter type: fast` or `slow Options:` *template-list*

The information can appear in any order. Not all the information has to be given. When you do not specify values for the items listed below, the values shown beside them are assigned by default.

Item	Default
`Command:`	(no default)
`Input types:`	any
`Output types:`	any
`Printer types:`	any
`Printers:`	any
`Filter type:`	`slow`
`Options:`	(no default)

As you can see, the default values define a very flexible filter, so you probably have to supply at least the input and output type(s). When you enter a list, you can separate the items in it with blanks or commas, unless it is a *templates-list;* items in a *templates-list* must be separated by commas.

Defining Options With Templates

A template is a statement in a filter definition that defines an option to be passed to the filter command based on the value of one of the characteristics of the filter. A filter definition may include more than one template. Multiple templates may be entered on a single line and separated with commas, or they may be entered on separate lines, preceded by the `Options:` prefix.

The format of a template is as follows:

keyword pattern = *replacement*

The *keyword* identifies the type of option being registered for a particular characteristic of the filter.

Let's look at an example of how an option is defined for a particular filter. Suppose you want to have the print service scheduler assign print requests to filters on the basis of the following criteria:

- If the type of `OUTPUT` to be produced by the filter is `impress`, then pass the `-I` option to the filter.

- If the type of `OUTPUT` to be produced by the filter is `postscript`, then pass the `-P` option to the filter.

To specify these criteria, provide the following templates as options to the `lpfilter` command.

```
Options: OUTPUT impress=-I, OUTPUT postscript=-P
```

NOTE If the `Options:` line becomes too long, put each template on a separate line, as follows:

```
Options: OUTPUT impress=-I
Options: OUTPUT postscript=-P
```

In both templates, the `keyword` is defined as `OUTPUT`. In the first template, the value of *pattern* is `impress` and the value of the *replacement* is `-I`. In the second template, the value of *pattern* is `postscript` and the value of the *replacement* is `-P`.

Template Keywords

The following thirteen *keywords* are available for defining `Options` in a filter definition:

Characteristic	*keyword*	Possible *patterns*	Example
Content type (input)	`INPUT`	*content-type*	`troff`
Content type (output)	`OUTPUT`	*content-type*	`postscript`
Printer type	`TERM`	*printer-type*	`att495`
Printer name	`PRINTER`	*printer-name*	`lp1`
Character pitch	`CPI`	*scaled-decimal*	`10`
Line pitch	`LPI`	*scaled-decimal*	`6`
Page length	`LENGTH`	*scaled-decimal*	`66`
Page width	`WIDTH`	*scaled-decimal*	`80`
Pages to print	`PAGES`	*page-list*	`1-5,13-20`
Character set	`CHARSET`	*character-set*	`finnish`
Form name	`FORM`	*form-name*	`invoice2`
Number of copies	`COPIES`	*integer*	`3`
Special modes	`MODES`	*mode*	`landscape`

To find out which values to supply for each type of template (that is, for the *pattern* and *replacement* arguments for each *keyword*), see the source of information listed below.

- The values for the `INPUT` and `OUTPUT` templates come from the file type that needs to be converted by the filter and the output type that has to be produced by the filter, respectively. They'll each be a type registered with the filter.
- The value for the `TERM` template is the printer type.
- The value for the `PRINTER` template is the name of the printer that will be used to print the final output.
- The values for the `CPI`, `LPI`, `LENGTH`, and `WIDTH` templates come from the user's request, the form being used, or the default values for the printer.
- The value for the `PAGES` template is a list of pages that should be printed. Typically, it is a comma separated list of page ranges, each of which consists of a dash separated pair of numbers or a single number (such as 1-5,6,8,10 for pages 1 through 5, 6, 8, and 10). However, whatever value was given in the `-P` option to a print request is passed unchanged.
- The value for the `CHARSET` template is the name of the character set to be used.
- The value for the `FORM` template is the name of the form requested by the `-f` option of the `lp` command.
- The value of the `COPIES` template is the number of copies that should be made of the file. If the filter uses this template, the LP print service will reduce to 1 the number of copies of the filtered file *it* will have printed, since this "single copy" will really be the multiple copies produced by the filter.
- The value of the `MODES` template comes from the `-y` option of the `lp` command (the command used to submit a print request). Because a user can specify several `-y` options, there may be several values for the `MODES` template. The values will be applied in the left-to-right order given by the user.

The *replacement* part of a template shows how the value of a template should be given to the filter program. It is typically a literal option, sometimes with the place-holder * included to show where the value goes. The *pattern* and *replacement* can also use the regular expression syntax of ed(1) for more complex conversion of user input options into filter options. All of the regular expression syntax of ed(1) is supported, including the `\( ... \)` and `\n` constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the `&`, which can be used to copy the entire *pattern* into the *replacement*.

NOTE If a comma or an equals sign (=) is included in a *pattern* or a *replacement*, escape its special meaning by preceding it with a backslash (\). Note that some regular expressions include commas that will have to be escaped this way. A backslash in front of any of these characters is removed when the *pattern* or *replacement* is used.

The following examples show how this works.

Example 1

You provide the following filter definition for a filter called `col`.

```
Input types:     N37, Nlp, simple
Output types:    simple
Command:         /usr/bin/col
Options:         TERM 450 = -b, MODES expand = -x
Options:         INPUT simple = -p -f
```

NOTE If you provide more than one definition (that is, more than one line) for any filter characteristic other than `OPTIONS`, only the second definition will be used by the print service.

After you have "registered" this definition with the print service by entering it as input with the `lpfilter` command, users' print requests will be handled as follows:

- If a user enters the command

```
lp -y expand report.dec10
```

the filter command will be run with the following arguments:

```
/usr/bin/col -x -p -f
```

- If a user enters the command

  ```
  lp -T N37 -y expand report.dec10
  ```

  the filter command will be run with the following arguments:

  ```
  /usr/bin/col -x
  ```

  Qualifier: The default printer is not of type 450.

- If a user enters the command

  ```
  lp -y expand -T 450 report.dec10
  ```

  the filter command will be run with the following arguments:

  ```
  /usr/bin/col -b -x
  ```

Example 2

The filter program is called `/usr/lib/lp/postscript/dpost`. It takes one input type, `troff`, produces an output type called `postscript`, and works with any printer of type `PS` (for PostScript). You've decided that your users need give just the abbreviations `port` and `land` when they ask for the paper orientation to be portrait mode and landscape mode, respectively. Because these options are not intrinsic to the LP print service, users must specify them using the `-y` option to the `lp` command.

The filter definition would look like this:

```
Input types: troff
Output types: postscript
Printer types: PS
Filter type: slow
Command: /usr/lib/lp/postscript/dpost
Options: LENGTH * = -l*
Options: MODES port = -pp, MODES land = -pl
```

A user submitting a file of type `troff` for printing on a PostScript printer (type `PS`), with requests for landscape orientation and a page length of 60 lines, would enter the following command:

```
lp -T troff -o length=60 -y land -d any
```

Then this filter would be invoked by the LP print service to convert the file as follows:

```
/usr/lib/lp/postscript/dpost -l60 -ol
```

Example 3

You add the following option template to the previous example:

```
Options:  MODES group\=\([1-9]\) = -n\1
```

This template is used to convert a `MODES` option of the form

`-y group=`*number*

into filter options

`-n`*number*

So if a user gives the following command

```
lp -y group=4
```

the `dpost` command would include the following options:

```
-n4
```

For additional examples, run the command

`lpfilter -f` *filter* `-l`

where *filter* is the name of the factory installed PostScript filters. (For a list of PostScript filters, see "PostScript Printers" later in this chapter.)

Command to Enter

Once a filter definition is complete, enter one of the following commands to add the filter to the system.

`lpfilter -f` *filter-name* `-F` *file-name* `lpfilter -f` *filter-name* `-`

The first command gets the filter definition from a file, and the second command gets the filter definition from the standard input. A *filter-name* can be any string you choose, with a maximum of fourteen alphanumeric characters and underscores.

If you need to change a filter, just reenter one of the same commands. You need only provide information for those items that must be changed; items for which you don't specify new information will stay the same.

Removing a Filter

The LP print service imposes no fixed limit on the number of filters you can define. It is a good idea, however, to remove filters no longer applicable, to avoid extra processing by the LP print service which must examine all filters to find one that works in a given situation.

To remove a filter, enter the following command:

```
lpfilter -f filter-name -x
```

Examining a Filter

Once you've added a filter definition to the LP print service, you can examine it by running the `lpfilter` command. The output of this command is the filter definition displayed in a format that makes it suitable as input. You may want to save this output in a file that you can use later to redefine the filter if you inadvertently remove the filter from the LP print service.

To examine a defined filter, enter one of the following commands:

```
lpfilter -f filter-name -l lpfilter -f filter-name -l >file-name
```

The first command presents the definition of the filter on your screen; the second command captures this definition in a file for future reference.

Restoring Factory Defaults

The software is shipped from the factory with a default set of filters. If, after changing them, you want to restore some or all of them, enter the following command:

```
        lpfilter -f filter-name -i
```

Replace *filter-name* with the name of the filter to restore, or the word `all` to restore all the default filters.

A Word of Caution

Adding, changing, or deleting filters can cause print requests still queued to be canceled. This is because the LP print service evaluates all print requests still queued, to see which are affected by the filter change. Requests that are no longer printable, because a filter has been removed or changed, are canceled (with notifications sent to the people who submitted them). There can also be delays in the responses to new or changed print requests when filters are changed, due to the many characteristics that must be evaluated for each print request still queued. These delays can become noticeable if there is a large number of requests that need to be filtered.

Because of this possible impact, you may want to make changes to filters during periods when the LP print service is not being used much.

Managing the Printing Load

Occasionally you may need to stop accepting print requests for a printer or move pending print requests from one printer to another. There are various reasons why you might want to do this, such as the following:

- the printer needs periodic maintenance
- the printer is broken
- the printer has been removed
- you've changed the configuration so that the printer is to be used differently
- too many large print requests are queued for one printer and should be spread around

If you are going to make a big change in the way a printer is to be used, such as stopping its ability to handle a certain form, changing the print wheels available for it, or disallowing some users from using it, print requests that are currently queued for printing on it will have to be moved or canceled. The LP print service will attempt to find alternate printers, but only if the user doesn't care which printer is to be used. Requests for a specific printer won't be automatically moved; if you don't move them first, the LP print service will cancel them.

If you decide to take a printer out of service, to change its configuration, or to lighten its load, you may want to move print requests off it and reject additional requests for it for awhile. To do so, use the `lpmove` and `reject` commands. If you do reject requests for a printer, you can accept requests for it later, by using the `accept` command.

Rejecting Requests for a Printer or Class

To stop accepting any new requests for a printer or class of printers, enter the following command.

```
reject -r "reason" printer-or-class-name
```

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* will be displayed whenever anyone tries to print a file on the printer. You can omit it (and the `-r`) if you don't want to specify a reason.

Although the `reject` command stops any new print requests from being accepted, it will not move or cancel any requests currently queued for the printer. These will continue to be printed as long as the printer is enabled.

Accepting Requests for a Printer or Class

After the condition that led to rejecting requests has been corrected or changed, enter one fo the the following commands to start accepting new requests.

```
accept printer-name
accept class-name
```

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 LP print service does not initially accept requests for new printers or classes.

Moving Requests to Another Printer

If you specify `-d any` when you run the `lp` command to queue a job, the print service schedules the job for a particular printer. If another becomes available first, the job is sent to the latter printer. If a job is scheduled for a given printer and you run `lpmove` to get jobs off that printer, that job will be moved off and the destination will change from any to the printer you've specified on the `lpmove` command line. Users may not have intended this side effect. If not, run the following command:

```
lp -i request-ID -d any
```

This command will change the destination for the requested job to the original destination: any (that is, any available printer).

If you have to move requests from one printer or class to another, enter one of the following commands

```
lpmove request-id printer-name lpmove printer-name1 printer-name2
```

You can give more than one request ID before the printer name in the first command.

The first command above moves the listed requests to the printer *printer-name*. The second command tries to move *all* requests currently queued for *printer-name*$_1$ to *printer-name*$_2$. If some requests cannot be printed on the new printer, they will be left in the queue for the original printer. When the second command is used, the LP print service also stops accepting requests for *printer-name*$_1$ (the same result you would obtain by running the command `reject` *printer-name*$_2$).

Examples

Here are some examples of how you might use these three commands:

Example 1

You've decided it is time to change the ribbon and perform some preventive maintenance on printer `lp1`. First, to prevent the loss of print requests already queued for `lp1`, you move all requests from printer `lp1` to printer `lp2`.

```
        lpmove lp1 lp2
```

After the requests are moved, make sure the LP print service does not print any more requests on `lp1` by disabling it.

```
        disable lp1
```

Now you may physically disable the printer and start working on it.

Example 2

You've finished changing the ribbon and doing the other work on `lp1`; now it's time to bring it back into service. Execute the following commands in any order:

```
        accept lp1 enable lp1
```

See the "Enabling and Disabling a Printer" section under "Making Printers Available" in this chapter.

Example 3

You notice that someone has queued several large files for printing on the printer laser1. Meanwhile laser2 is idle because no one has queued requests for it. Move the two biggest requests (laser1-23 and laser1-46) to laser2, and reject any new requests for laser1 for the time being.

```
lpmove laser1-23 laser1-46 laser2 reject -r "too
busy--will reopen later" laser1
```

Managing Queue Priorities

The LP print service provides a simple priority mechanism that users can use to adjust the position of a print request in the queue. Each print request can be given a priority level by the user who submits it; this is a number from 0 to 39, with *smaller* numbers indicating *higher* levels of priority. Requests with higher priority (smaller numbers) are placed ahead of requests with lower priority (larger numbers).

Thus, for example, a user who decides that her print request is of low priority can assign it a larger value when she submits the file for printing. Another user who decides that his print request is of high priority can assign it a smaller value when he submits the file for printing.

A priority scheme this simple would not work if there were no controls on how high one can set the priority. You can define the following characteristics in 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 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 LP print service give "immediate" handling to a print request, and you can have it put on "hold" another print request. This will allow the first request to be printed and will delay the second print request until you allow it to be "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 to move it up for immediate printing, respectively. These commands are discussed in detail below.

Setting Priority Limits

To set a user's priority limit, enter the following command.

```
lpusers -q priority-level -u user-name
```

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, though). The argument *priority-level* is a number from 0 to 39. As mentioned before, the lower number the higher the priority, or, in this case, the priority limit.

If you want to set a priority limit for the remaining users, enter the following command:

```
lpusers -q priority-level
```

This sets the default limit; the default applies to those users for whom you have not set a personal limit, using the first `lpusers` command.

If you later decide that someone should have a different priority limit, just reenter the first command above with a new limit. Or, if you decide that the default limit is more appropriate for someone who already has a personal limit, enter the following command:

```
lpusers -u user-name
```

Again, you can do this for more than one user at a time by including a list of names. Using the `lpusers` command with just the `-u` option removes users' personal priority limits and puts the "default limit" into effect for those users.

Setting a Default Priority

To set the default priority (the priority level assigned to print requests submitted without a priority), use the following command:

```
lpusers -d priority-level
```

Don't confuse this default with the "default limit." This default is applied when a user doesn't specify a priority level; the default limit is applied if you haven't assigned a limit for a user—it is used to limit the user from requesting 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 LP print service will use a 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.

```
lpusers -l
```

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:

- you can adjust the priority to any level, regardless of the limit for the user (who may adjust it only up to his or her limit)
- both you and the user can put it on hold and allow other requests to be printed ahead of it
- you can put it at the head of the queue for immediate printing

Use the `lp(1)` command to do any of these tasks.

Changing the Priority for a Request

If you want to change the priority of a particular request that is still waiting to be printed, you can assign a new priority level to it. By doing so, you can move it in the queue so that it is ahead of lower priority requests, and behind requests at the same level or of higher priority. The priority limit assigned to the user (or the default priority limit) has no effect because, as the administrator, you can override this limit.

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.

Putting a Request on Hold

Any request that has not finished printing can be put on hold. This will stop its printing, if it is currently printing, and keep it from printing until you resume it. A user may also put his or her own request on hold and then resume it, but may not resume a print request that has been put on hold by the administrator.

To place a request on hold, enter the following command:

```
lp -i request-ID -H hold
```

Enter the following command to resume the request:

```
lp -i request-ID -H resume
```

Once resumed a request will continue to move up the queue and will eventually be printed. If printing had already begun when you put it on hold, it will be the next request printed.

Moving a Request to the Head of the Queue

You can move a print request to the head of the queue where it will be the next one eligible for printing. If you want it to start printing immediately but another request is currently being printed, you may interrupt the first request by putting it on hold, as described above.

Enter the following command to move a print request to the head of the queue:

```
lp -i request-ID -H immediate
```

Only you, as the administrator, can move a request in this way; regular users cannot use the `-H immediate` option.

NOTE

If you set more than one request for immediate printing, the requests will be printed in the reverse order set; that is, the request moved to the head of the queue most recently will be printed first.

Starting and Stopping the LP Print Service

Under normal operation, you should never have to start or stop the LP print service manually. It is automatically started each time the UNIX system is started, and stopped each time the UNIX system is stopped. If, however, you need to stop the LP print service without stopping the UNIX system as well, you can do so by following the procedure described below.

Stopping the LP print service will cause all printing to cease within seconds. Any print requests that have not finished printing will be printed in their entirety after the LP print service is restarted. The printer configurations, forms, and filters in effect when the LP print service is stopped will be restored after it is restarted.

NOTE To start and stop the LP print service manually, you must be logged in as either the user `lp` or the super-user (`root`).

Manually Stopping the Print Service

To stop the LP print service manually, enter the following command:

```
lpshut
```

The message

```
Print services stopped.
```

will appear, and all printing will cease within a few seconds. If you try to stop the LP print service when it is not running, you will see the message

```
Print services already stopped.
```

Manually Starting the Print Service

To restart the LP print service manually, enter the following command:

```
lpsched
```

The message

```
Print services started.
```

will appear. It may take a minute or two for the printer configurations, forms, and filters to be reestablished, before any saved print requests start printing. If you try to restart the LP print service when it is already running, you will see the message

```
Print services already active.
```

NOTE The LP print service does not have to be stopped to change printer configurations or to add forms or filters.

Directories and Files Used by the LP Print Service

This section lists the directories and files used by the LP print service. You can use this list to see if any files are missing or if the ownership or access permissions have changed. Normal operation of the LP print service should not cause any problems. However, if you do notice any discrepancies, there may be a security breach on your system.

At the end of this section is a description of the script used to clean out the request log (`/var/lp/logs/requests`) periodically. You may want to change this script to have the file cleaned out more or less frequently, or to condense the information into a report. See "Cleaning Out the Request Log" later in this section.

The various LP print service files and directories are found under the main directories listed below. These main directories should have the access permissions and ownerships shown.

Permissions	Owner	Group	Directory or File
`drwxrwxr-x`	`lp`	`lp`	`/var/spool/lp`
`drwxrwxr-x`	`lp`	`lp`	`/var/lp`
`drwxrwxr-x`	`lp`	`lp`	`/etc/lp`
`drwxr-xr-x`	`root`	`other`	`/usr/lib/lp`

You can check this by entering the following command:

```
ls -ld /var/spool/lp /var/lp /etc/lp /usr/lib/lp
```

Under these directories you should see only the files and directories shown in the table on the next few pages. You can generate a similar table for comparison by entering this command:

```
ls -lR /var/spool/lp /var/lp /etc/lp /usr/lib/lp
```

Permissions	Owner	Group	Directory or File
`/var/spool/lp:`			
`-rw-rw-r--`	`lp`	`lp`	`SCHEDLOCK`
`drwxrwxr-x`	`lp`	`lp`	`admins`
`lrwxrwxrwx`	`lp`	`lp`	`bin -> /usr/lib/lp/bin`
`lrw-rw-r--`	`lp`	`lp`	`default -> /etc/lp/default`
`drwxrwxr-x`	`lp`	`lp`	`fifos`
`lrwxrwxr-x`	`lp`	`lp`	`logs -> /var/lp/logs`
`lrwxrwxr-x`	`lp`	`lp`	`model -> /usr/lib/lp/model`
`drwxrwxr-x`	`lp`	`lp`	`requests`
`drwxrwxr-x`	`lp`	`lp`	`system`
`lrwxrwxrwx`	`lp`	`lp`	`temp -> /var/spool/lp/tmp/sfsti`
`drwx--x--x`	`lp`	`lp`	`tmp`
`-rw-r--r--`	`lp`	`lp`	`users -> /etc/lp/users`
`/var/spool/lp/admins:`			
`lrwxrwxrwx`	`lp`	`lp`	`lp -> /etc/lp`
`/var/spool/lp/fifos:`			
`prw-rw-rw-`	`lp`	`lp`	`FIFO`
`prw-------`	`root`	`other`	`listenBSD`
`prw-------`	`root`	`other`	`listenS5`
`drwxrwx--x`	`lp`	`lp`	`private`
`drwxrwx-wx`	`lp`	`lp`	`public`
`/var/spool/lp/fifos/private:`			
`pr--------`	*user*	*group*	*systemPID*
⋮			
`/var/spool/lp/fifos/public:`			
`pr--------`	*user*	*group*	*systemPID*
⋮			
`/var/spool/lp/requests:`			
`drwxrwx---`	`lp`	`lp`	*system1*

Permissions	Owner	Group	Directory or File
drwxrwx---	lp	lp	*system2*
⋮			
drwxrwx---	lp	lp	*systemN*
/var/spool/lp/requests/*systemK*:			
-rw-rw----	lp	lp	*id1*-0
-rw-rw----	lp	lp	*id2*-0
⋮			
-rw-rw----	lp	lp	*idN*-0
/var/spool/lp/system:			
-rw-rw-r--	lp	lp	cstatus
-rw-rw-r--	lp	lp	pstatus
/var/spool/lp/tmp:			
drwxrwxr-x	lp	lp	*system1*
drwxrwxr-x	lp	lp	*system2*
⋮			
drwxrwxr-x	lp	lp	*systemN*
/var/spool/lp/tmp/*systemK*:			
-rw-------	lp	lp	*idN*-0
-rw-------	lp	lp	*idN*-1
-rw-------	lp	lp	*idN*-2
⋮			
-rw-------	lp	lp	*idN*-*M*
-rw-------	lp	lp	F*idN*-1
-rw-------	lp	lp	F*idN*-2
⋮			
-rw-------	lp	lp	F*idN*-*M*
-rw-------	lp	lp	*idN*
-rw-------	lp	lp	A-*K*

Permissions	Owner	Group	Directory or File
-rw-------	lp	lp	F-*K*
-rw-------	lp	lp	P-*K*
/var/lp/logs:			
-rw-rw----	lp	lp	lpsched
-rw-rw----	lp	lp	requests
-rw-rw-r--	root	other	lpNetLog
-rw-rw-r--	root	other	p*PID*
-rw-rw-r--	root	other	c*PID*
/etc/lp:			
-rw-rw-r--	lp	lp	Systems
drwxrwxr-x	lp	lp	classes
-rw-rw-r--	lp	lp	filter.table
-rw-rw-r--	lp	lp	filter.table.i
drwxrwxr-x	lp	lp	forms
drwxrwxr-x	lp	lp	interfaces
lrwxrwxrwx	lp	lp	logs -> /var/lp/logs
drwxrwxr-x	lp	lp	printers
drwxrwxr-x	lp	lp	printwheels
-rw-rw-r--	lp	lp	users
/etc/lp/classes:			
-rw-rw-r--	lp	lp	*class1*
-rw-rw-r--	lp	lp	*class2*
⋮			
-rw-rw-r--	lp	lp	*classN*
/etc/lp/forms:			
drwxrwxr-x	lp	lp	*form1*
drwxrwxr-x	lp	lp	*form2*
⋮			
drwxrwxr-x	lp	lp	*formN*

Permissions	Owner	Group	Directory or File
`/etc/lp/forms/`*formK*`:`			
`-rwxrwx---`	`lp`	`lp`	`alert.sh`
`-rw-rw----`	`lp`	`lp`	`alert.vars`
`-rw-rw----`	`lp`	`lp`	`align_ptrn`
`-rw-rw-r--`	`lp`	`lp`	`allow`
`-rw-rw-r--`	`lp`	`lp`	`comment`
`-rw-rw-r--`	`lp`	`lp`	`deny`
`-rw-rw-r--`	`lp`	`lp`	`describe`
`/etc/lp/interfaces:`			
`-rwxrwxr-x`	`lp`	`lp`	*printer1*
`-rwxrwxr-x`	`lp`	`lp`	*printer2*
⋮			
`-rwxrwxr-x`	`lp`	`lp`	*printerN*
`/etc/lp/printers:`			
`drwxrwxr-x`	`lp`	`lp`	*printer1*
`drwxrwxr-x`	`lp`	`lp`	*printer2*
⋮			
`drwxrwxr-x`	`lp`	`lp`	*printerN*
`/etc/lp/printers/`*printerK*`:`			
`-rwxrwx---`	`lp`	`lp`	`alert.sh`
`-rw-rw----`	`lp`	`lp`	`alert.vars`
`-rw-rw-r--`	`lp`	`lp`	`comment`
`-rw-rw-r--`	`lp`	`lp`	`configuration`
`-rw-rw-r--`	`lp`	`lp`	`forms.allow`
`-rw-rw-r--`	`lp`	`lp`	`forms.deny`
`-rw-rw-r--`	`lp`	`lp`	`residentfonts`
`-rw-rw-r--`	`lp`	`lp`	`users.allow`
`-rw-rw-r--`	`lp`	`lp`	`users.deny`
`/etc/lp/pwheels:`			
`drwxrwxr-x`	`lp`	`lp`	*printwheel1*

Permissions	Owner	Group	Directory or File
`drwxrwxr-x`	`lp`	`lp`	*printwheel2*
⋮			
`drwxrwxr-x`	`lp`	`lp`	*printwheelN*
`/etc/lp/pwheels/`*printwheelK*`:`			
`-rwxrwx---`	`lp`	`lp`	`alert.sh`
`-rw-rw----`	`lp`	`lp`	`alert.vars`
`/usr/lib/lp:`			
`dr-xr-xr-x`	`lp`	`lp`	`bin`
`---x--x--x`	`lp`	`lp`	`lpNet`
`---x--x---`	`lp`	`lp`	`lpdata`
`---s--x--x`	`root`	`lp`	`lpsched`
`drwxrwxr-x`	`lp`	`lp`	`model`
`/usr/lib/lp/bin:`			
`-r--r--r--`	lp	`lp`	`alert.proto`
`-r-xr-xr-x`	lp	`lp`	`drain.output`
`-r-xr-xr-x`	lp	`lp`	`lp.cat`
`-r-xr-xr-x`	lp	`lp`	`lp.set`
`-r-xr-xr-x`	lp	`lp`	`lp.tell`
`-r-xr-xr-x`	lp	`lp`	`slow.filter`
`/usr/lib/lp/model:`			
`-rwxrwxr-x`	`lp`	`lp`	`standard`

The italicized names—*printerN*, *formN*, *classN*, *printwheelN*, *idN*, and *systemN*—are placeholders for a single printer, form, class, print wheel, request ID, and UNIX system name, respectively (*idN* is just the numeric part of the request ID.) There will be one set of these directories and files for each active printer, form, class, print wheel and request configured on your system, and for each system used for remote printing. The italicized letter *K* is a placeholder for an internal number; the `A-`*K*, `F-`*K*, and `P-`*2K*, files are used to store alert messages.

The ownership and permissions of the *idN-M* request files under the `/var/spool/lp/tmp/`*system* directory will change during the life of a print request, alternating between the user who submitted the request and the `lp` ID.

The two directories under the `/var/spool/lp/fifos` directory contain named pipes used to communicate between the LP print service and commands such as `lpadmin`, `lpstat`, and `lp`. These directories must have the permission flags and ownership shown if communication with the LP print service is to work. Every entry below these directories is given a unique name formed by combining the name of the system (the node name) and the process ID of the command. The uniqueness of the entry names prevents two or more people from accidentally sharing the same communications path.

Cleaning Out the Request Log

The directories `/var/spool/lp/tmp/`*system* and `/var/spool/lp/requests/`*system* contain files that describe each request that has been submitted to the LP 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 `/var/spool/lp/requests/`*system* directory where it can be kept secure: the request file in the `/var/spool/lp/tmp/`*system* is safe from all except the user who submitted the request, while the file in `/var/spool/lp/requests/`*system* is safe from all users, including the submitting user.

These files remain in their directories only as long as the request is in the queue. Once the request is finished, the information in the files is combined and appended to the file `/var/lp/logs/requests`. This file is not removed by the LP print service, but can be cleaned out periodically, using, for instance, the `cron` facility. (See the description of the `crontab` command in the *User's Reference Manual.*)

The default `crontab` entry provided with the LP print service is shown below.

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

(This is one line in the `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 to the `agefile` program keeps 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 to be saved more often, or if you want the file to be cleaned out more often than once a day, you can change the time when the `crontab` entry is run by changing the first two numbers. The current values, 13 and 3, cause cleaning up to be done at 3:13 A.M. each day.

The default `crontab` entry supplied is sufficient to keep the old print request records from accumulating in the spooling file system. You may want to condense information in the request log to produce a report on the use of the LP print service, or to aid in generating accounting information. You can produce a different script that examines the file and extracts information just before the clean up procedure.

The request log has a simple structure that makes it easy to extract data from it using common UNIX shell commands. Requests are listed in the order they are printed, and are separated by lines showing their request IDs. Each line below the separator line is marked with a single letter that identifies the kind of information contained in that line. Each letter is separated from the data by a single space. See the following table for details.

Letter	Content of line
=	This is the separator line. It contains 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.

Letter	Content of line
	The user ID, group ID, and sizes are preceded by the words `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; files were printed in the order given.
`f`	The name of the form used.
`H`	One of three types of special handling: `resume`, `hold`, and `immediate`. The only useful value found in this line will be `immediate`.
`N`	The type of alert used when the print request was 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 included 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, shown as a combination of individual bits expressed in hexadecimal form. While several bits are used internally by the print service, the most important bits are listed below: `0x0004` Slow filtering finished successfully. `0x0010` Printing finished successfully.

Letter	Content of line
	`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 will differ 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 LP print service administrator.

PostScript Printers

PostScript is a general purpose programming language, like C or Pascal. In addition to providing the usual features of a language, however, PostScript allows a programmer to specify the appearance of both text and graphics on a page.

A PostScript printer is a printer equipped with a computer that runs an interpreter for processing PostScript language files. When a PostScript printer receives a file, it runs that file through the interpreter and then prints it. Unless special provisions have been made by the manufacturer, files submitted to a PostScript printer must be written in the PostScript language.

Why would you want to use a PostScript printer? PostScript provides excellent facilities for managing text and graphics and combining them. Graphics operators facilitate the construction of geometric figures which can then be positioned and scaled with any orientation. The text capabilities allow the user to specify a number of different fonts that can be placed on a page in any position, size, or orientation. Because text is treated as graphics, text and graphics are readily combined. Moreover, the language is resolution and device independent, so that draft copies can be proofed on a low-resolution device and the final version printed in higher resolution on a different device.

Applications that support PostScript, including word-processing and publishing software, will create documents in the PostScript language without intervention by the user. Thus, it is not necessary to know the details of the language to take advantage of its features. However, standard files that many applications produce cannot be printed on a PostScript printer because they are not described in the language. The LP print service provides optional filters to convert many of these files to PostScript so that users may take advantage of PostScript and continue to use their standard applications, such as `troff`.

How to Use a PostScript Printer

When the PostScript printers and filters have been installed, LP manages PostScript files like any others. If `psfile` is a file containing a PostScript document and psprinter has been defined to LP as a PostScript printer, the command

```
lp -d psprinter -T postscript psfile
```

will schedule the print request and manage the transmission of the request to the PostScript printer.

Support of Non-PostScript Print Requests

Because PostScript is a language and PostScript printers are expecting print requests written in that language, some applications may produce standard print requests that may not be intelligible to PostScript printers. The following are examples of print requests that may not be interpreted by some PostScript printers.

Content Type	Type of Print Request
`troff`	Print output from the `troff` command.
`simple`	Print an ASCII ("simple") text file.
`dmd`	Print the contents of a bit-mapped display from a terminal such as an AT&T 630.
`tek4014`	Print files formatted for a Tektronix 4014 device.
`daisy`	Print files intended for a Diablo 630 ("daisy-wheel") printer.
`plot`	Print plot-formatted files.

Filters are provided with the LP print service to translate print requests with these formats to the PostScript language. For example, to convert a file containing ASCII text to PostScript code, the filter takes that text and writes a program around it, specifying printing parameters such as fonts and the layout of the text on a page.

Once the PostScript filters are installed, they will be invoked automatically by the LP print service when a user specifies a content-type for a print request with the `-T` option. For example, if a user enters the command

```
lp -d psprinter -T simple report2
```

the ASCII file `report2` (a file with an ASCII or "simple" format) will be converted to PostScript automatically, as long as the destination printer (`psprinter`) has been defined to the system as a PostScript printer.

Additional PostScript Capabilities Provided by Filters

The filters previously described also take advantage of PostScript capabilities to provide additional printing flexibility. Most of these features may be accessed through the "mode option" (invoked by the `-y` option) to the `lp` command. These filters allow you to use several unusual options for your print jobs. The following list describes these options and shows the option you should include on the `lp` command line for each one.

`-y reverse`	Reverse the order in which pages are printed.
`-y landscape`	Change the orientation of a physical page from portrait to landscape.
`-y x=`*number*`,y=`*number*	Change the default position of a logical page on a physical page by moving the origin.
`-y group=`*number*	Group multiple logical pages on a single physical page.
`-y magnify=`*number*	Change the logical size of each page in a document.
`-o length=`*number*	Select the number of lines in each page of the document.
`-P` *number*	Select, by page numbers, a subset of a document to be printed.
`-n` *number*	Print multiple copies of a document.

NOTE If these filters are to be used with an application that creates PostScript output, make sure that the format of the application conforms to the format of the PostScript file structuring comments. In particular, the beginning of each PostScript page must be marked by the comment

`%%Page:` *label ordinal*

where *ordinal* is a positive integer that specifies the position of the page in the sequence of pages in the document, and *label* is an arbitrary page label.

For example, say you have a file called `report2` that has a content type `simple` (meaning that the content of this file is in ASCII format). You want to print six pages of this file (pages 4-9) with two logical pages on each physical page. Because one of the printers on your system (`psprinter`) is a PostScript printer, you can do this by entering the following command:

```
lp -d psprinter -T simple -P 4-9 -y group=2 myfile
```

The filter which groups these logical pages will try to position the pages on the physical page to maximize space utilization. Thus when you specify `group=2`, the pages will be printed side by side, so that the physical page will be landscape orientation. Landscape mode, which controls the orientation of the logical page rather than the physical page, would cause the logical pages to be positioned one on top of the other when combined with the `group=2` option.

The Administrator's Duties

Support of PostScript printers is similar to support of other printers, in that the printers must be defined to the system with the `lpadmin` command and the appropriate software must be installed to manage them. PostScript printers may require some additional effort in supporting fonts.

Installing and Maintaining PostScript Printers

PostScript printers, like other printers, are installed with the `lpadmin` command.

NOTE The printer-type and content-type of a PostScript printer must be consistent with the printer type used in PostScript filters. Therefore you should install your PostScript printers with a printer-type of PS or PSR, and a content-type of PS.

The printer types PS and PSR serve two functions. First, they cause LP to activate the `postio` filter to communicate with the printer. Second, the standard interface shell creates a PostScript banner page for printers with printer type PS or PSR. The banner page is printed last if the printer-type is PSR, and the pages of the document are printed in reverse order. The printer type is specified with the `-T` option in the `lpadmin` command.

As part of the installation, you may want to install fonts on the printer or downloadable fonts on the computer. See "Installing and Maintaining PostScript Fonts" later in this chapter for details.

Installing and Maintaining PostScript Filters

PostScript filters are provided with UNIX System V Release 4 and are installed during regular installation. This installation covers the majority of situations. In certain circumstances, however, you may find it helpful to change the filter descriptions and install the filters differently. To help you do this, this section provides describes the location and function of these filters.

PostScript filters are contained in the directory

```
/usr/lib/lp/postscript
```

NOTE There are two types of filters: fast filters and slow filters. For definitions of these types, see `lpfilter`(1M) in the *System Administrator's Reference Manual* and "Defining a Filter" earlier in this chapter.

A prerequisite of communication between any system and a PostScript printer is the presence of the `postio` filter on the system. This program is the only mandatory PostScript filter that communicates directly with the PostScript printer. The following filters allow other types of documents to be translated to PostScript and to be printed on a PostScript printer.

File Content Type	Filter
`simple`	`postprint`
`troff`	`dpost`
`daisy`	`postdaisy`
dmd(AT&T 630)	`postdmd`
`tek4014`	`posttek`
`plot`	`postplot`

The following filters perform special functions:

Function	Filter
Communicate with printer	`postio`
Download fonts	`download`
Reverse or select pages	`postreverse`
Matrix gray scales	`postmd`

Installing and Maintaining PostScript Fonts

One of the advantages of PostScript is its ability to manage fonts. Fonts are stored in outline form, either on the printer or on a computer that communicates with a printer. When a document is printed, the PostScript interpreter generates each character as needed (in the appropriate size) from the outline description of it. If a font required for a document is not stored on the printer being used, it must be transmitted to that printer before the document can be printed. This transmission process is called "downloading fonts."

Fonts are stored and accessed in several ways.

- Fonts may be stored permanently on a printer. These "printer resident" fonts may be installed in ROM on the printer by the manufacturer. If the printer has a disk, fonts may be installed on that disk by you (that is, by the print service administrator). Most PostScript printers are shipped with thirty-five standard fonts.
- A font may be "permanently-downloaded" by being transmitted to a printer with a PostScript "exitserver" program. A font downloaded in this way will remain in the printer's memory until the printer is turned off. Memory allocated to this font will reduce the memory available for PostScript print requests. Use of exitserver programs requires the printer system password and may be reserved for the printer administrator. This method is useful when there is continual use of a font by the majority of print requests serviced by that printer.
- Fonts may be prepended to a user's print request by the user, and be transmitted as part of the user's print request. When the user's document has been printed, the space allocated to the font is freed for other print requests. The font is stored in the user's directory. This method is preferable for fonts with more limited usage.

- Fonts may be stored on a system shared by many users. These fonts may be described as "host-resident." This system may be a server for the printer or may be a system connected to the printer by a network. Each user may request fonts in the document to be printed. This method is useful when there are a large number of available fonts or when there is not continual use of these fonts by all print requests. If the fonts will be used only on printers attached to a server, they should be stored on the server. If the fonts are to be used by users on one system, who may send jobs to multiple printers on a network, they may be stored on the users' system.

The LP print service allows you to manage fonts in any of these ways. It provides a special download filter to manage fonts using the last method described above.

The LP print service supplies `troff` width tables for the thirty-five standard PostScript fonts which reside on many PostScript printers, for use by the `troff` program.

Managing Printer-Resident Fonts

Most PostScript printers come equipped with fonts resident in the printer ROM. Some printers have a disk on which additional fonts are stored. When a printer is installed, the list of printer-resident fonts should be added to the font-list for that printer. These lists are kept in the printer administration directories. For a particular printer, this list is contained in the file

`/etc/lp/printers/`*printer-name*`/residentfonts`

where *printer-name* is the name of the printer. When fonts are permanently downloaded to the printer, the font names should be added to this file. This will prevent fonts from being downloaded when they are already on the printer, a time-consuming procedure. If the printer is attached to a remote system, this list should include fonts which reside on that system and are available for downloading to the printer. This prevents fonts from being transmitted unnecessarily across a network. These files must be edited manually; that is, with the help of a text editor such as `vi`.

Installing and Maintaining Host-Resident Fonts

Some fonts will be resident on the host and transmitted to the printer as needed for particular print requests. As the administrator, it's your job to make PostScript fonts available to all the users on a system. To do so, you must know how and where to install these fonts, using the guidelines described previously. Because fonts are requested by name and stored in files, the LP print service keeps a map file that shows the correspondence between the names of fonts and the names of the files containing those fonts. Both of these must be updated when fonts are installed on the host.

Install host-resident PostScript fonts by doing the following:

- Copy the font file to the appropriate directory.
- Add to the map table the name of the font and the name of the file in which it resides.
- If you are using `troff`, you must create new width tables for this font in the standard `troff` font directory.

Where Are Fonts Stored?

The fonts available for use with PostScript printers reside in directories called `/usr/share/lib/hostfontdir/`*typeface*/*font* where *typeface* is replaced by a name such as `palatino` or `helvetica`, and *font* is replaced by a name such as `bold` or `italic`.

Adding an Entry to the Map Table

Also within the `hostfontdir` directory, you (the administrator) must create and maintain a map table that shows the correspondence between the name assigned to each font by the foundry (the company that created the font) and the name of the file in which that font resides. For example, to map the font called "Palatino Bold," add the following line to the map table:

```
Palatino-Bold /usr/share/lib/hostfontdir/palatino/bold
```

(The map table itself is in the file `/usr/share/lib/hostfontdir/map`).

Once this entry exists in the map table on your system, your users will be able to have a Palatino Bold font used in their print jobs. When they submit, for printing, a file containing a request for this font, the LP print service will prepend a copy of the file

```
/usr/share/lib/hostfontdir/palatino/bold
```

to that file before sending it to the printer.

Downloading Host-Resident Fonts

The creators of the PostScript language anticipated that users would want to download fonts to printers. The *PostScript Language Reference Manual* (by Adobe Systems, Inc., Addison-Wesley Publishing Co., Inc., 1985) states the following:

> "...programs that manage previously generated PostScript page descriptions, such as 'printer spooler' utilities, may require additional information about those page descriptions. For example, if a page description references special fonts, a spooler may need to transmit definitions of those fonts to the PostScript printer ahead of the page description itself.
>
> To facilitate these and other operations, [PostScript] defines a standard set of *structuring conventions* for PostScript programs."

The download filter relies on these structuring conventions to determine which fonts must be downloaded.

When the LP PostScript document contains a request for fonts not loaded on the printer, the download filter manages this request. This filter is invoked as a "fast filter"; it downloads fonts automatically if the fonts reside on the same system as the printer. The download filter may also be used to send fonts to a remote printer. To do this, you may create a new filter table entry which calls the download filter as a "slow" filter through the `-y` option. Alternatively, you may force selection of this filter by changing the input-type.

The download filter does five things:

- It searches the PostScript document to determine which fonts have been requested. These requests are documented with the following PostScript structuring comments:

  `%%DocumentFonts:` *font1 font2 ...*

  in the header comments.

- It searches the list of fonts resident on that printer to see if the requested font must be downloaded.
- If the font is not resident on the printer, it searches the host-resident font directory (by getting the appropriate file name from the map table) to see if the requested font is available.
- If the font is available, the filter takes the file for that font and prepends it to the file to be printed.
- The filter sends the font definition file and the source file (the file to be printed) to the PostScript printer.

Customizing the Print Service

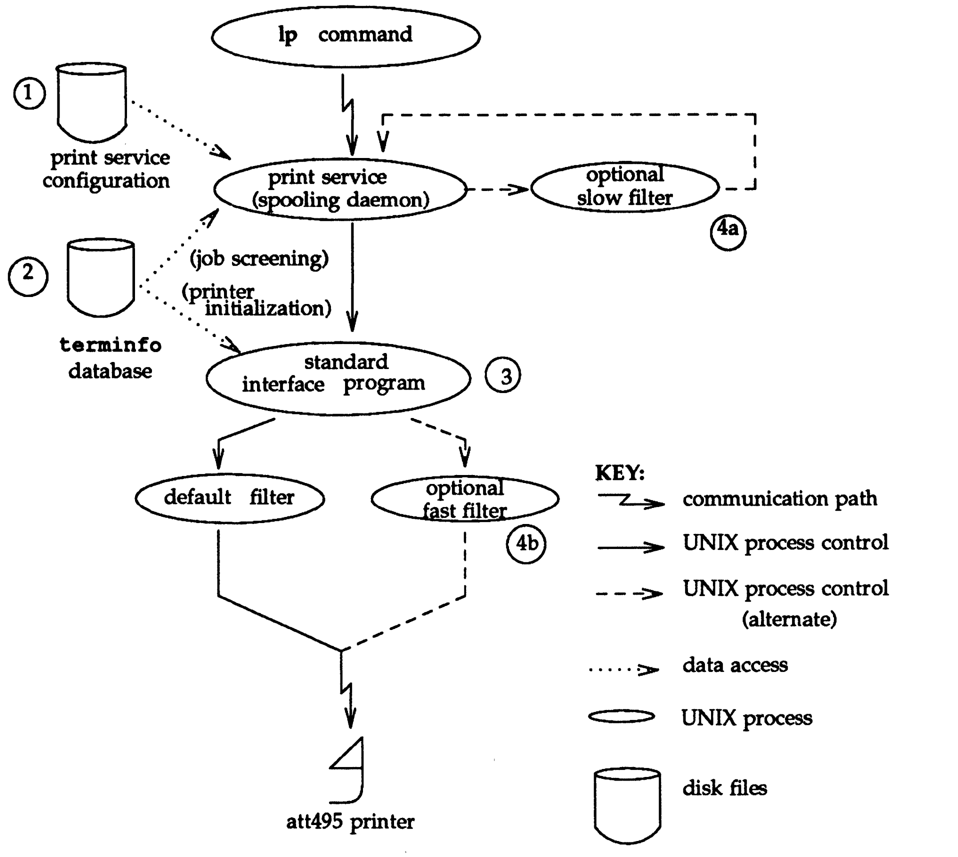
Although the LP print service has been designed to be flexible enough to handle most printers and printing needs, it doesn't handle every possible situation. You may buy a printer that doesn't quite fit into the way the LP print service handles printers, or you may have a printing need that the standard features of the LP print service don't accommodate.

You can customize the LP 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, and
- write a filter.

The diagram in Figure 9-6 gives an overview of the processing of a print request.

Figure 9-6: How LP Processes Print Request `lp -d att495` *file*



Each print request is sent to a "spooling daemon" that keeps track of all requests. The daemon is created when you start the LP print service. This UNIX system process is also responsible for keeping track of the status of printers and slow filters; when a printer finishes printing a user's file, the daemon starts it printing another request (if there is one queued).

To customize the print service, adjust or replace some of the pieces shown in Figure 9-6. (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 explain how to do this. Configuration data that are relatively dependent on the printer include the printer port characteristics: baud rate, parity, and so on.

2. For a printer that is not represented in the `terminfo` database, you can add a new entry that describes its capabilities. The `terminfo` 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 in a state where it is ready to print a request.

   For instance, if the `terminfo` database does not contain an entry for a printer capable of setting a page length requested by a user, the spooling daemon will reject the request. On the other hand, if it does contain an entry for such a printer, then the same information will be 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 LP 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 a user's files to the printer.

4a. and 4b.
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 LP print service match the printer communication settings. The standard printer port settings have been designed to work with typical files and many printers, but they won't work with all files and printers. This isn't really a customizing step, because a standard feature of the LP print service is to allow you to specify the port settings for each printer. However, it's an important step in getting your printer to work with the LP print service, so it's 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 LP print service). Then read the manual page for the `stty`(1) command in the *User's Reference Manual*. It summarizes the various characteristics that can be set on a terminal or printer port.

Only some of the characteristics listed in the `stty`(1) manual page are important for printers. The ones likely to be of interest to you are listed below (but you should still consult the `stty`(1) manual page for others).

`stty` Option	Meaning
`evenp`	Send even parity in the 8th bit
`oddp`	Send odd parity in the 8th bit
`-parity`	Don't generate parity; send all 8 bits unchanged
`110 - 38400`	Set the communications speed to this baud rate
`ixon`	Enable XON/XOFF (also known as START/STOP or DC1/DC3) flow control
`-ixon`	Turn off XON/XOFF flow control
`-opost`	Don't do any "output post-processing"
`opost`	Do "output post-processing" according to the settings listed below

`onlcr`	Send a carriage return before every linefeed
`-onlcr`	Don't send a carriage return before every linefeed
`ocrnl`	Change carriage returns into linefeeds
`-ocrnl`	Don't change carriage returns into linefeeds
`-tabs`	Change tabs into an equivalent number of spaces
`tabs`	Don't change tabs into spaces

When you have a set of printer port characteristics you think should apply, adjust the printer configuration as described in the section "How to Define Printer Ports and Printer Port Characteristics" under "Printer Management" in this chapter. You may find that the default settings are sufficient for your printer.

Adjusting the `terminfo` Database

The LP print service relies on a standard interface and the `terminfo` database to initialize each printer and establish 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 LP print service. Several entries for AT&T printers and other popular printers are delivered in the standard `terminfo` database.

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`. The "Acceptable Terminal Names" section of Appendix G ("Setting Up the Terminal") in the *User's Guide* describes how to determine a correct `TERM` variable for a user's terminal; you can use it as a guide for picking a known name for your printer.

If you cannot find a `terminfo` entry for your printer, you should add one. If you don't, you may still be able to use the printer with the LP print service but you won't have the option of 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 LP print service uses fewer than 50 of these. The following table lists the items that need to be defined (as appropriate for the printer) to add a new printer to the LP print service.

`terminfo` item	Meaning
Booleans:	
`cpix`	Changing character pitch changes resolution
`daisy`	Printer needs operator to change character set
`lpix`	Changing line pitch changes resolution
Numbers:	
`bufsz`	Number of bytes buffered before printing
`cols`	Number of columns in a line
`cps`	Average print rate in characters per second
`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
Strings:	
`chr`	Change horizontal resolution
`cpi`	Change number of characters per inch
`cr`	Carriage return
`csnm`	List of character set names
`cud1`	Down one line
`cud`	Move carriage down # lines
`cuf`	Move carriage right # columns
`cuf1`	Carriage right
`cvr`	Change vertical resolution

`terminfo` item	Meaning
`ff`	Page eject
`hpa`	Horizontal position absolute
`ht`	Tab to next 8-space tab stop
`if`	Name of initialization file
`iprog`	Path name of initializing program
`is1`	Printer initialization string
`is2`	Printer initialization string
`is3`	Printer initialization string
`lpi`	Change number of lines per inch
`mgc`	Clear all margins (top, bottom, and sides)
`rep`	Repeat a character # times
`rwidm`	Disable double wide printing
`scs`	Select character set
`scsd`	Start definition of a character set
`slines`	Set page length to # lines per page
`smgl`	Set left margin at current column
`smglp`	Set left margin
`smgr`	Set right margin at current column
`smgrp`	Set right margin
`smglr`	Set both left and right margins
`smgt`	Set top margin at current line
`smgtp`	Set top margin
`smgb`	Set bottom margin at current line
`smgbp`	Set bottom margin
`smgtb`	Set both top and bottom margins
`swidm`	Enable double wide printing
`vpa`	Vertical position absolute

To construct a database entry for a new printer, see details about the structure of the `terminfo` database in the `terminfo`(4) manual page (*UNIX System V Programmer's Reference Manual*).

Once you've made the new entry, you need to compile it into the database using the `tic`(1) program (available in the Terminal Information Utilities). Just enter the following command:

`tic` *file-name*

file-name is the name of the file containing the `terminfo` entry you have crafted for the new printer.

NOTE The LP 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 LP print service so it can read the new information.

How to Modify the Interface Program

NOTE If you have an interface program that you have used with the LP Spooling Utilities before UNIX System V Release 3.2, it should still work with the LP print service. Note, though, that several **-o** options have been "standardized," and will be passed to every interface program. These may interfere with similarly named options used by your interface.

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, and change it to fit, rather than starting from scratch. You can find a copy of it under the name

```
/var/spool/lp/model/standard
```

What Does the Interface Program Do?

Any interface program is responsible for doing the following tasks:

- Initializing the printer port, if necessary. The generic interface program uses the `stty` command to do this.
- Initializing the physical printer. The generic interface program uses the `terminfo` database and the `TERM` shell variable to get the control sequences to do this.
- Printing a banner page, if necessary.
- Printing the correct number of copies of the request content.

An interface program is not responsible for opening the printer port. The LP print service opens the port, a process which includes calling a "dial-up" printer, if one is used to connect the printer. The printer port connection is given to the interface program as standard output, and the printer is identified as the "controlling terminal" for the interface program so that a "hang-up" of the port will cause a `SIGHUP` signal to be sent to the interface program.

A customized interface program must not terminate the connection to the printer or "uninitialize" the printer in any way.

How Is the Interface Program Used?

When the LP print service routes an output request to a printer, the interface program for the printer is invoked as follows:

```
/var/spool/lp/admins/lp/interface/P ID user title copies \
        options file1 file2 ...
```

Arguments for the interface program are:

P	printer name
id	request ID returned by the `lp(1)` command
user	logname of the user who made the request
title	optional title specified by the user
copies	number of copies requested by the user
options	blank-separated list of options specified by the user or set by the LP print service
file$_1$, *file*$_2$, ...	full pathnames of the files 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 blank-separated 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-number*$_1$
`lpi=`*decimal-number*$_2$

These options specify a format of *decimal-number*$_1$ columns per inch and *decimal-number*$_2$ 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 *decimal-number*$_1$ and are synonyms, respectively, for `10` columns per inch, `12` columns per inch, and as many columns per inch as possible.

`length=`*decimal-number*$_1$
`width=`*decimal-number*$_2$

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 *stty-option-list* as a set of 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.

`lpd='`*argument-list*'

This option is used internally by the `lpsched` command; you can ignore it.

`flist='`*file-list*'

This option is used internally by the `lpsched` command; you can ignore it.

The above options may be specified by the user when issuing a print request. Alternatively, they may be specified by the LP print service from defaults given by the administrator either 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 the following 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 `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 and 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 either the LP print service or the user with a line such as 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 below.

```
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. The way that this is set will vary, 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 in the code for the standard interface program for the section 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 will be mailed to the user; the messages printed on the standard output will end up on the printed page where they can be read by the user when he or she picks up the output.

When printing is complete, your interface program should exit with a code that shows the status of the print job. Exit codes are interpreted by the LP print service as follows:

Code	Meaning to the LP print service
0	The print request has been completed successfully. If a printer fault has occurred, it has been cleared.
1 to 127	A problem has been encountered in printing this particular request (for example, too many non-printable characters, or the request exceeds the printer capabili-

	ties). The LP print service notifies the person who submitted the request that there was an error in printing it. This problem will not affect future print requests. If a printer fault had occurred, it has been cleared.
128	Reserved for internal use by the LP print service. Interface programs must not exit with this code.
129	A printer fault has been encountered in printing the request. This problem will affect future print requests. If the fault recovery for the printer directs the LP print service to wait for the administrator to fix the problem, the LP print service will disable the printer. If the fault recovery is to continue printing, the LP print service will not disable the printer, but will try printing again in a few minutes.
greater than 129	These codes are reserved for internal use by the LP 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 LP print service has no choice but to reprint the request from the beginning when the fault has been cleared. Another way of getting an alert to the administrator (that does not require the entire request to be reprinted) is to have the interface program send a fault message to the LP print service but wait for the fault to clear. When the fault clears, the interface program can resume printing the user's file. When the printing is finished, the interface program can give a zero exit code just as if the fault had never occurred. An added advantage is that the interface program can detect when the fault is cleared automatically, so that the administrator doesn't have to enable the printer.

Fault messages can be sent to the LP print service using the `lp.tell` program. This is referenced using the `$LPTELL` shell variable in the standard interface code. The program takes its standard input and sends it to the LP print service where it is put into the message that alerts the administrator to the printer fault. If its standard input is empty, `lp.tell` does not initiate an alert. Examine the standard interface code immediately after these comments for an example of how the `lp.tell` (`$LPTELL`) program is used:

```
# Here's where we set up the $LPTELL program to cap-
ture # fault messages.

# Here's where we print the file.
```

If the special exit code 129 or the `lp.tell` program is used, there is no longer a need for the interface program to disable the printer itself. Your interface program can disable the printer directly, but doing so will override the fault alerting mechanism. Alerts are sent only if the LP print service detects the printer has faulted, and the special exit code and the `lp.tell` program are its main detection tools.

If the LP print service has to interrupt the printing of a file at any time, it will "kill" the interface program with a signal `TERM` (trap number 15; see `kill(1)` and `signal(2)` in the *UNIX System V User's Reference Manual* and *UNIX System V Programmer's Reference Manual*, respectively). If the interface program dies from receipt of any other signal, the LP print service assumes that future print requests won't be affected, and continues to use the printer. The LP print service notifies the person who submitted the request that the request has not been finished successfully.

When the interface is first invoked, the signals `HUP`, `INT`, `QUIT`, and `PIPE` (trap numbers 1, 2, 3, and 13) are ignored. The standard interface changes this so that these signals are trapped at appropriate times. The standard interface interprets receipt of these signals as warnings that the printer has a problem; when it receives one, it issues a fault alert.

How to Write a Filter

A filter is used by the LP print service each time it has to print a type of file that isn't acceptable by a printer. A filter can be as simple or as complex as needed; there are only a few external requirements:

- The filter should get the content of a user's file from its standard input and send the converted file to the standard output.

- A `slow` filter can send messages about errors in the file to standard error. A `fast` filter should not, as described below. Error messages from a `slow` filter are collected and sent to the user who submitted the file for printing.
- If a `slow` filter dies because of receiving a signal, the print request is stopped and the user who submitted the request is notified. Likewise, if a `slow` filter exits with a non-zero exit code, the print request is stopped and the user is notified. The exit codes from `fast` filters are treated differently, as described below.
- A filter should not depend on other files that normally would not be accessible to a regular user; if a filter fails when run directly by a user, it will fail when run by the LP print service.

The "Filter Management" section earlier in this chapter describes how to add a filter to the LP print service.

If you want your filter to detect printer faults, you must also fulfill the following requirements:

- If possible, the filter should wait for a fault to be cleared before exiting. Additionally, it should continue printing at the top of the page where printing stopped after the fault clears. If the administrator does not want this contingency followed, the LP print service will stop the filter before alerting the administrator.
- It should send printer fault messages to its standard error as soon as the fault is recognized. It does not have to exit, but can wait as described above.
- It should *not* send messages about errors in the file to standard error. These should be included in the standard output stream, where they can be read by the user.
- It should exit with a zero exit code if the user's file is finished (even if errors in the file have prevented it from being printed correctly).
- It should exit with a non-zero exit code *only* if a printer fault has prevented it from finishing a file.
- When added to the filter table, it must be added as a `fast` filter. (See the "Defining a Filter" section in this chapter for details.)

Quick Reference to LP Print Service Administration

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 either as `root` or as `lp`. (`lp` is a system login, use of which requires a password.) For a description of how to set up a password for a system login, see the "Security" chapter.

You'll also probably need to use the commands for disabling and enabling a printer and the rest of the user commands.

- Activating a printer:

  `enable`(1)
- Canceling a request for a file to be printed:

  `cancel`(1)
- Sending a file (or files) to a printer:

  `lp`(1)
- Reporting the status of the LP print service:

  `lpstat`(1)
- Deactivating a specified printer(s):

  `disable`(1)
- Permitting job requests to be queued for a specific destination:

  `/usr/sbin/accept`(1M)
- Preventing jobs from being queued for a specified destination:

  `/usr/sbin/reject` – described on the `accept`(1M) manual page
- Setting up or changing printer configurations:

  `/usr/sbin/lpadmin`(1M)
- Setting up or changing filter definitions:

  `/usr/sbin/lpfilter`(1M)

- Setting up or changing preprinted forms:

  /usr/sbin/lpforms(1M)

- Mounting a form:

  /usr/sbin/lpadmin(1M)

- Moving output requests from one destination to another:

  /usr/sbin/lpmove – described on the lpsched(1M) manual page

  See lpsched(1M).

- Starting the LP print service scheduler:

  /usr/lib/lp/lpsched(1M)

- Stop the LP print service scheduler

  /usr/sbin/lpshut(1M) – described on the lpsched(1M) manual page

- Setting or changing the default priority and priority limits that can be requested by users of the LP print service:

  /usr/sbin/lpusers(1M)

10 Process Scheduling

Introduction

The UNIX system scheduler determines when processes run. It maintains process priorities based on configuration parameters, process behavior, and user requests; it uses these priorities to assign processes to the CPU.

System V Release 4 gives users absolute control over the order in which certain processes run and the amount of time each process may use the CPU before another process gets a chance.

By default, the Release 4 scheduler uses a time-sharing policy like the policy used in previous releases. A time-sharing policy adjusts process priorities dynamically in an attempt to provide good response time to interactive processes and good throughput to processes that use a lot of CPU time.

The System V Release 4 scheduler offers a real-time scheduling policy as well as a time-sharing policy. Real-time scheduling allows users to set fixed priorities on a per-process basis. The highest-priority real-time user process always gets the CPU as soon as it is runnable, even if system processes are runnable. An application can therefore specify the exact order in which processes run. An application may also be written so that its real-time processes have a guaranteed response time from the system.

For most UNIX environments, the default scheduler configuration works well and no real-time processes are needed: administrators should not change configuration parameters and users should not change scheduler properties of their processes. However, when the requirements for an application include strict timing constraints, real-time processes sometimes provide the only way to satisfy those constraints..

NOTE Real-time processes used carelessly can have a dramatic negative effect on the performance of time-sharing processes.

This chapter is addressed to administrators of systems that include the System V Release 4 scheduler. There are at least two reasons why administrators should understand the scheduler:

- The scheduler has an overriding effect on the performance and perceived performance of a system. The default scheduler is tuned to perform well in representative work environments, but you must understand how it operates to know whether you can reconfigure it to better suit local needs.

- A bug in a real-time program or a malicious real-time user can lock out all other processing, including kernel processing. Users need root permission to create real-time processes, and presumably only trustworthy users have this permission. But administrators still should be aware that the scheduler functions introduce new ways to cause trouble, and should be prepared for accidents and for misuse of these functions.

For programming information on the scheduler, see the *Programmer's Guide: System Services and Programming Support Tools*. The primary user command for controlling process scheduling is `priocntl`(1), which is described in the *User's Reference Manual*. The primary function call for controlling process scheduling is `priocntl`(2), which is described in the *Programmer's Reference Manual*.

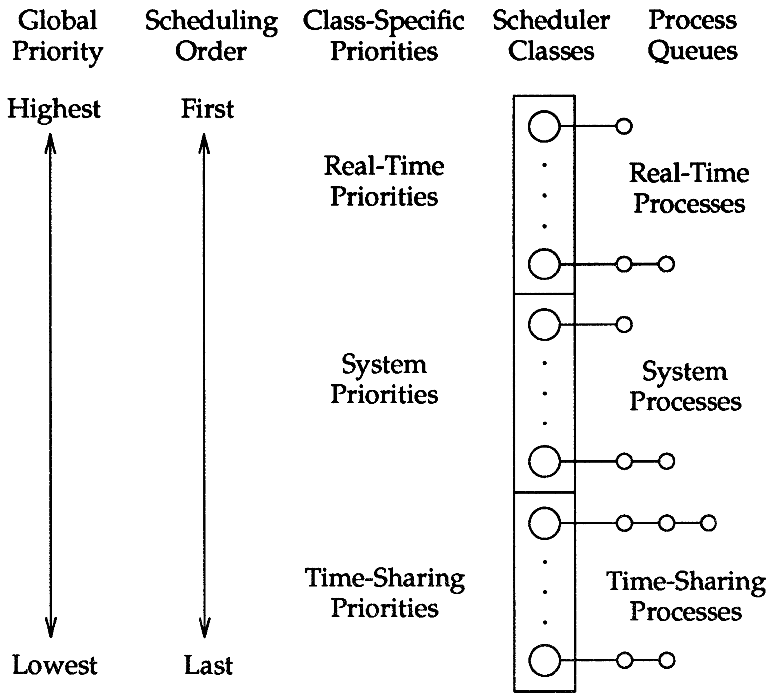
The rest of this chapter is organized as follows:

- The "Overview of the Process Scheduler" tells what the scheduler does and how it does it. It also introduces scheduler classes.
- "Configuring the Scheduler" describes how you can control the scheduler using tunable parameters and the two scheduler parameter tables: `ts_dptbl`(4) for time-sharing parameters and `rt_dptbl`(4) for real-time parameters.
- "Changing Scheduler Parameters with `dispadmin`" tells how to display or change scheduler parameters in a running system. `dispadmin` changes do not survive a reboot. To make permanent changes in scheduler configuration, you must change the scheduler parameter tables in the `master.d` directory.

Overview of the Process Scheduler

The following figure shows how the System V Release 4 process scheduler works:

Figure 10-1: The System V Release 4 Process Scheduler



When a process is created, it inherits its scheduler parameters, including scheduler class and a priority within that class. A process changes class only as a result of a user request. The system manages the priority of a process based on user requests and a policy associated with the scheduler class of the process.

In the default configuration, the initialization process belongs to the time-sharing class. Because processes inherit their scheduler parameters, all user login shells begin as time-sharing processes in the default configuration.

The scheduler converts class-specific priorities into global priorities. The global priority of a process determines when it runs—the scheduler always runs the runnable process with highest global priority. Numerically higher priorities run first. Once the scheduler assigns a process to the CPU, the process runs until it uses up its time slice, sleeps, or is preempted by a higher-priority process. Processes with the same priority run round-robin.

Administrators specify default time slices in the configuration tables, but users may assign per-process time slices to real-time processes.

You can display the global priority of a process with the `-cl` options of the `ps(1)` command. You can display configuration information about class-specific priorities with the `priocntl(1)` command and the `dispadmin(1M)` command.

By default, all real-time processes have higher priorities than any kernel process, and all kernel processes have higher priorities than any time-sharing process.

NOTE As long as there is a runnable real-time process, no kernel process and no time-sharing process runs.

The following sections describe the scheduling policies of the three default classes.

Time-Sharing Class

The goal of the time-sharing policy is to provide good response time to interactive processes and good throughput to CPU-bound processes. The scheduler switches CPU allocation frequently enough to provide good response time, but not so frequently that it spends too much time doing the switching. Time slices are typically on the order of a few hundred milliseconds.

The time-sharing policy changes priorities dynamically and assigns time slices of different lengths. The scheduler raises the priority of a process that sleeps after only a little CPU use (a process sleeps, for example, when it starts an I/O operation such as a terminal read or a disk read); frequent sleeps are characteristic of interactive tasks such as editing and running simple shell commands. On the other hand, the time-sharing policy lowers the priority of a process that uses the CPU for long periods without sleeping.

The default time-sharing policy gives larger time slices to processes with lower priorities. A process with a low priority is likely to be CPU-bound. Other processes get the CPU first, but when a low-priority process finally gets the CPU, it gets a bigger chunk of time. If a higher-priority process becomes runnable during a time slice, however, it preempts the running process.

The scheduler manages time-sharing processes using configurable parameters in the time-sharing parameter table `ts_dptbl`. This table contains information specific to the time-sharing class.

System Class

The system class uses a fixed-priority policy to run kernel processes such as servers and housekeeping processes like the paging demon. The system class is reserved for use by the kernel; users may neither add nor remove a process from the system class. Priorities for system class processes are set up in the kernel code for those processes; once established, the priorities of system processes do not change. (User processes running in kernel mode are not in the system class.)

Real-Time Class

The real-time class uses a fixed-priority scheduling policy so that critical processes can run in predetermined order. Real-time priorities never change except when a user requests a change. Contrast this fixed-priority policy with the time-sharing policy, in which the system changes priorities in order to provide good interactive response time.

Privileged users can use the `priocntl` command or the `priocntl` system call to assign real-time priorities.

The scheduler manages real-time processes using configurable parameters in the real-time parameter table `rt_dptbl`. This table contains information specific to the real-time class.

Configuring the Scheduler

The default configuration includes both the time-sharing and the real-time scheduler classes. The time-sharing class is tuned for representative UNIX system workloads. Such workloads have a high proportion of interactive processes, which sleep early and often. The real-time class is configured for applications that need it.

For traditional time-sharing uses such as software development, office applications, and document production, real-time processes may be unnecessary. In addition, they may be undesirable. First, they consume memory that cannot be paged: the u-blocks of real-time processes are never paged out. Second, they introduce new ways to cause performance problems: a high-priority real-time process can block out all other processing. In a computing environment where only time-sharing is needed, you may want to remove the real-time scheduler class from your configuration, as described below in the section "Changing Scheduler Configuration."

On the other hand, if a machine is running an application that has strict requirements on the order in which processes must run, then the real-time scheduler class provides the only way to guarantee that those requirements are met.

NOTE Real-time processes can have a dramatic negative effect on time-sharing performance.

This section describes the parameters and tables that control scheduler configuration, and tells how to reconfigure the scheduler. A basic assumption is that your workload is reasonable for your system resources, such as CPU power, primary memory, and I/O capacity. If your workload does too much computation or too much I/O for your hardware, reconfiguring the scheduler won't help. See the "Performance Management" chapter for more information.

Default Global Priorities

The following table shows the scheduling order and global priorities for each scheduler class.

Scheduling Order	Global Priority	Scheduler Class
first	159	
	.	
	.	Real-Time
	.	
	100	
	99	
	.	
	.	System
	.	
	60	
	59	
	.	
	.	Time-Sharing
	.	
last	0	

When your system is built, it constructs this information from the tunable parameters and scheduler parameter tables described in the following sections. Although you are not forced to configure scheduler classes to produce consecutive, non-overlapping global priorities like the default priorities, we recommend that you do so for the sake of simplicity. Likewise, we recommend that you make all real-time global priorities greater than the global priorities of all other classes. These conventions simplify scheduler configuration, and they should be able to accommodate any requirements on the scheduler.

Kernel processes such as the swapper and the paging demon run in the system scheduler class. Kernel processes must compete with user processes for CPU time, and in the default configuration all real-time processes have higher priorities than all system processes. Therefore, real-time applications must be written carefully to ensure that the kernel gets the processing time it needs. Also, if you reconfigure the scheduler, make sure that the system class gets enough priority over the time-sharing class to give kernel processes the CPU time they need.

Tunable Parameters

This section describes the tunable parameters that control scheduler configuration. These parameters are specified in files in the /etc/master.d directory.

The following parameters are specified in the kernel file:

- MAXCLSYSPRI is the maximum global priority of processes in the system class. When the kernel starts system processes, it assigns their priorities using MAXCLSYSPRI as a reference point.

  NOTE MAXCLSYSPRI must be 39 or greater, because the kernel assumes it has at least that great a range of priorities below MAXCLSYSPRI. If you request a MAXCLSYSPRI below 39, it is changed to 39.

  The most important system processes get global priorities at or near MAX-CLSYSPRI; the least important system processes get global priorities at or near (MAXCLSYSPRI - 39). The default value of MAXCLSYSPRI is 99, which gives all system processes higher priorities than all user processes.

- INITCLASS is the scheduler class assigned to the init process. This scheduler class is inherited by all descendants of init, which normally include all user login shells. By default, INITCLASS is TS; that is, all login shells are time-sharing processes in the default configuration.

- SYS_NAME is the character string name of the system scheduler class. The default value of SYS_NAME is SYS.

The following parameters are specified in the ts file, which controls the time-sharing policy:

- TSMAXUPRI specifies the range within which users may adjust the priority of a time-sharing process using the priocntl system call: the valid range is –TSMAXUPRI to +TSMAXUPRI. The default value of TSMAXUPRI is 20. (Configuring a value of 20 emulates the behavior of the older, less general scheduler interfaces nice and setpriority, which continue to work as in previous releases.)

  The value of TSMAXUPRI is independent of the configured number of global time-sharing priorities, though we recommend configuring at least 40 time-sharing priorities, as explained below in the section on ts_dptbl. In the default configuration, there are 60 time-sharing priorities, but users may adjust their priorities only within a range of −20 to +20. The system may use the remaining priorities depending on process behavior.

- NAMETS specifies the character string name of the time-sharing scheduler class. This name is returned by the priocntl system call and it is assigned to the tunable parameter INITCLASS to specify the default scheduler class for user processes. The default value of NAMETS is TS.

The following parameter is specified in the rt file, which controls the real-time policy:

- NAMERT specifies the character string name of the real-time scheduler class. The default value of NAMERT is RT.

Real-Time Parameter Table rt_dptbl

The scheduler uses rt_dptbl(4), the real-time scheduler (or dispatcher) parameter table, to manage real-time processes. A default version of rt_dptbl is delivered with the system, and an administrator may change it to suit local needs. rt_dptbl is specified in the rt file in the master.d directory. It is built into the kernel as part of system configuration if the system file contains the line

```
INCLUDE:RT
```

You may adjust the size and values of rt_dptbl depending on the applications on your system. Here is part of a simple rt_dptbl:

```
rt_glbpri  rt_qntm
      100,     100,
      101,      80,
      102,      60,
      103,      40,
      104,      20,
      105,      10
```

- The rt_glbpri column contains global priorities (the priorities that determine when a process runs). Higher numbers run first.
- The rt_qntm column contains the default time slice (or quantum) associated with the priority in the rt_glbpri column; this is the maximum amount of time a process with this priority may use the CPU before the scheduler gives another process a chance. This time slice is specified in clock ticks. (The system clock ticks HZ times per second, where HZ is a hardware-dependent constant defined in the param.h header file.)

The highest priority specified in this table is 105, so processes with priority 105 always run before any other processes. If it does not sleep, a process with priority 105 runs for 10 clock ticks before the scheduler looks for another process to run. (Because 105 is the highest priority, a process at this priority would be preempted after its time slice only if there were another process with priority 105.) Processes at priority 104 run for 20 clock ticks, and so on. The lowest real-time priority specified in this table is 100; a process with priority 100 runs for 100 clock ticks.

The default real-time priority is the lowest priority configured in rt_dptbl. This is the priority assigned to a process if it is changed to a real-time process and no priority is specified. This is also the priority assigned to the init process and all its children if INITCLASS is set to RT.

Though rt_dptbl contains default time slices for real-time priorities, users with the appropriate privilege can set real-time priority and time slice independently. Users can specify any time slice they want for a real-time process, including an infinite time slice. The system assumes that real-time processes voluntarily give up the CPU so other work can get done.

Time-Sharing Parameter Table ts_dptbl

The scheduler uses ts_dptbl(4), the time-sharing scheduler (or dispatcher) parameter table, to manage time-sharing processes. A default version of ts_dptbl is delivered with the system, and an administrator may change it to suit local needs. Save a backup of the default version of ts_dptbl. ts_dptbl is specified in the ts file in the master.d directory. It is automatically built into the kernel as part of system configuration.

You may change the size and values of ts_dptbl depending on your local needs, but only experienced administrators should make such changes. The default values have a long history of good performance over a wide range of environments. Changing the values is not likely to help much, and inappropriate values can have a dramatic negative effect on system performance.

If you do decide to change ts_dptbl, we recommend that you include at least 40 time-sharing global priorities. A range this large gives the scheduler enough leeway to distinguish processes based on their CPU use, which it must do to give good response to interactive processes. The default configuration has 60 time-sharing priorities. Here is part of a simple ts_dptbl:

glbpri	qntm	tqexp	slprt	mxwt	lwt
0,	100,	0,	1,	5,	1,
1,	90,	0,	2,	5,	2,
2,	80,	1,	3,	5,	3,
3,	70,	1,	4,	5,	4,
4,	60,	2,	5,	5,	5,
5,	50,	2,	6,	5,	6,
6,	40,	3,	7,	5,	7,
7,	30,	3,	8,	5,	8,
8,	20,	4,	9,	5,	9,
9,	10,	4,	9,	5,	9,

- The glbpri column contains global priorities (the priorities that determine when a process runs). Higher numbers run first.

  In the table above, the global priorities run from a high of 9 to a low of 0.

- The qntm column contains the time slice (or quantum) associated with the priority in the glbpri column; this is the maximum amount of time a process with this priority may use the CPU before the scheduler gives another process a chance. This time slice is specified in clock ticks. (The system clock ticks HZ times per second, where HZ is a hardware-dependent constant defined in the /usr/include/sys/param.h header file.)

  In the table above, time slices run from 10 clock ticks for the highest-priority processes to 100 clock ticks for the lowest-priority processes.

- The `tqexp` column determines the new process priority for a process whose time slice expires before it sleeps. If a process at the priority in the `glbpri` column uses its whole time slice without sleeping, the scheduler changes its priority using the `tqexp` column as an index back into `ts_dptbl`: the new priority is the global priority in the `tqexp` position in `ts_dptbl`. (In the default configuration, the index of an entry in `ts_dptbl` happens to match the global priority of that entry. However, this match is not necessary.)

  It is usually reasonable to lower the priority of a time-sharing process whose time slice expires, because the process is too CPU-bound for its current priority. A long, CPU-intensive process is an extreme example of such a process, and its priority should usually be lowered in favor of processes that sleep after a little CPU use, which are more likely to be interactive processes.

  In the table above, process priorities are cut roughly in half when a time slice expires. The lowest priority (0) stays at 0, priority 1 is reduced to 0, priorities 2 and 3 are reduced to 1, and so on.

- The `slprt` column gives the priority assigned to a process when it returns from a sleep. A process may sleep voluntarily, as it does when it makes certain system calls, or involuntarily, as it does when the kernel puts it to sleep after a page fault, for example. It is usually reasonable to raise the priority of a process that has slept. It has shown desirable behavior (it gave up the CPU), so we reward it by giving it a higher priority when it awakes.

  In the table above, process priorities are incremented by 1 after they sleep, except that the highest time-sharing priority (9) stays the same.

- The `mxwt` column specifies the number of seconds a process can remain runnable before having its priority changed. (The priority is changed using the `lwt` column. See the explanation below.)

  In the table above, all priorities are recalculated after a wait of 5 seconds.

- The `lwt` column gives the new priority for a process that is runnable for `mxwt` seconds without getting its full time slice. It is usually reasonable to raise the priority of a process that is not getting any CPU time.

  In the table above, process priorities are incremented by 1 when they have been runnable for 5 clock ticks, except that the highest priority time-sharing priority (9) stays the same.

The default global priority of a time-sharing process is the priority in the middle of ts_dptbl. This is the priority assigned to a process if it is changed to a time-sharing process with default parameters. This is also the priority initially assigned to the init process if INITCLASS is set to TS. The descendants of init, normally including all login shells and other user processes, inherit its class and current scheduler parameters.

Kernel-Mode Parameter Table ts_kmdpris

The scheduler uses the kernel-mode parameter table ts_kmdpris to manage sleeping time-sharing processes. A default version of ts_kmdpris is delivered with the system, and there is seldom a reason to change it. ts_kmdpris is specified in the ts file in the master.d directory. It is automatically built into the kernel as part of system configuration.

NOTE The kernel assumes that it has at least 40 priorities in ts_kmdpris. It panics if it does not.

The kernel-mode parameter table is a one-dimensional array of global priorities. The kernel assigns these priorities to sleeping processes based on their reasons for sleeping. If a user process sleeps because it is waiting for an important resource, such as an inode, it sleeps at a priority near the high end of the ts_kmdpris priorities, so that it may get and free the resource quickly when the resource becomes available. If a user process sleeps for a less important reason, such as a wait for terminal input, it sleeps at a priority near the low end of the ts_kmdpris priorities.

The default kernel-mode parameter table is simply a one-dimensional array of the integers from 60 through 99, which means that time-sharing processes sleep at priorities between the default real-time priorities and the default time-sharing priorities.

In the default configuration, the priorities in ts_kmdpris happen to be exactly the same as the priorities used by system class processes, because the tunable parameter MAXCLSYSPRI is the same as the highest priority in ts_kmdpris. This overlap is designed to be consistent with the scheduler behavior of previous releases of the UNIX system, because these priorities produce good performance in most environments. But the overlap is not necessary. The System V

Release 4 scheduler introduces a logical separation between the priorities of system processes and sleeping time-sharing processes; an administrator may configure a machine so that the two sets of processes have different ranges of global priorities.

Changing Scheduler Configuration

Changing scheduler configuration requires changing one or more of the tunable parameters or the configuration tables `rt_dptbl`, `ts_dptbl`, and `ts_kmdpris`. You can change any of these by changing the appropriate file in the `master.d` directory and rebuilding the kernel as described in the "Performance Management" chapter. Changes made in this way are permanent. This is the only way to change the size of the configuration tables.

See the section below on the `dispadmin` command for a way to make a temporary change on a running system.

Removing a Scheduler Class

For systems that do not need real-time processes, it may make sense to remove the real-time class, thereby making it impossible to create real-time processes. By not having real-time processes, you avoid their non-pageable u-blocks and you avoid the possibility of a runaway process monopolizing the machine. To remove the real-time scheduler class:

- Replace the `INCLUDE:RT` line from the `/etc/system` file with

```
EXCLUDE:RT
```

- Rebuild the kernel.

There should be no reason to remove the time-sharing scheduler class. An application can put its crucial processes into the real-time class, and thereby ensure that they always run before any time-sharing process. However, if you have a compelling reason to remove the time-sharing class, here's how to do it:

- Replace the `INCLUDE:TS` line from the `/etc/system` file with:

```
EXCLUDE:TS
```

- In the `kernel` file in `master.d`, change `INITCLASS` to `RT`. This makes `init` and all its descendants real-time processes.
- Rebuild the kernel.

Installing a Scheduler Class

By default, both the time-sharing and the real-time scheduler classes are installed. Therefore, you need to install a class only if you first remove it.

To install the time-sharing class:

- Make sure that the `TS` module is in the `/boot` directory.
- Insert the `INCLUDE:TS` line in the `system` file. (The `TS` module is automatically configured unless it is explicitly `EXCLUDE`d.)
- Build the kernel.

To install the real-time class:

- Make sure that the `RT` module is in the `/boot` directory.
- Insert the `INCLUDE:RT` line is in the `system` file. (The `RT` module is not configured unless it is explicitly `INCLUDE`d.)
- Build the kernel.

When you re-install a scheduler class, you should also check the value of the tunable parameter `INITCLASS` to make sure that your configuration is assigning the default scheduler class you want.

Changing Scheduler Parameters with `dispadmin`

The dispadmin(1M) command allows you to change or retrieve scheduler information in a running system. Changes made using dispadmin do not survive a reboot. To make permanent configuration changes, you must change the scheduler parameter tables in the master.d directory as described in the section above on configuration. However, you can use dispadmin to get an effect equivalent to changing configuration tables by calling dispadmin from a startup script that changes the configuration automatically at boot time.

The dispadmin command has three forms:

- dispadmin -l lists the configured scheduler classes.
- dispadmin -g [-r *res*] -c *class* gets scheduler parameters for the specified class. By default, time slices are printed in milliseconds. You may optionally retrieve time slices at a resolution specified by the -r option.
- dispadmin -s *config_file* -c *class* sets scheduler parameters for the specified class from *config_file* (your configuration file).

Here is the output of the -l option for the default configuration.

```
$ dispadmin -l
CONFIGURED CLASSES
==================

SYS     (System Class)
TS      (Time Sharing)
RT      (Real Time)
```

The -g option gets current scheduler parameters for the specified class and writes them to the standard output. Scheduler parameters are class specific. The parameters for the default classes are described in the sections above on the scheduler parameter tables; ts_dptbl holds time-sharing parameters and rt_dptbl hold real-time parameters.

The following screen shows part of the output of `dispadmin -g` for the real-time class:

```
$ dispadmin -c RT -g      # list real-time parameters
# Real Time Dispatcher Configuration
RES=1000

# TIME QUANTUM                    PRIORITY
# (rt_quantum)                      LEVEL
      1000                  #          0
      1000                  #          1
      1000                  #          2
      1000                  #          3
      1000                  #          4
      1000                  #          5
      1000                  #          6
      1000                  #          7
      1000                  #          8
      1000                  #          9
       800                  #         10
       800                  #         11
       800                  #         12
       800                  #         13
       800                  #         14
       800                  #         15
       800                  #         16
       800                  #         17
       800                  #         18
       800                  #         19
       ...                           ...
       100                  #         50
       100                  #         51
       100                  #         52
       100                  #         53
       100                  #         54
       100                  #         55
       100                  #         56
       100                  #         57
       100                  #         58
       100                  #         59
```

The following screen shows part of the output of `dispadmin -g` for the time-sharing class:

```
$ dispadmin -c TS -g     # list time-sharing parameters
# Time Sharing Dispatcher Configuration
RES=1000

# ts_quantum  ts_tqexp  ts_slpret  ts_maxwait ts_lwait  PRIORITY LEVEL
       1000         0        10          5        10        #     0
       1000         0        11          5        11        #     1
       1000         1        12          5        12        #     2
       1000         1        13          5        13        #     3
       1000         2        14          5        14        #     4
       1000         2        15          5        15        #     5
       1000         3        16          5        16        #     6
       1000         3        17          5        17        #     7
       1000         4        18          5        18        #     8
       1000         4        19          5        19        #     9
        800         5        20          5        20        #    10
        800         5        21          5        21        #    11
        800         6        22          5        22        #    12
        800         6        23          5        23        #    13
        800         7        24          5        24        #    14
        800         7        25          5        25        #    15
        800         8        26          5        26        #    16
        800         8        27          5        27        #    17
        800         9        28          5        28        #    18
        800         9        29          5        29        #    19
        ...       ...       ...        ...       ...             ...
        100        40        55          5        55        #    50
        100        41        55          5        55        #    51
        100        42        56          5        56        #    52
        100        43        56          5        56        #    53
        100        44        57          5        57        #    54
        100        45        57          5        57        #    55
        100        46        58          5        58        #    56
        100        47        58          5        58        #    57
        100        48        59          5        59        #    58
        100        49        59          5        59        #    59
```

By default, `dispadmin` reports time slices in milliseconds. If you specify the `-r` *res* option, `dispadmin` reports time slices in units of *res* intervals per second. For example, a *res* of 1000000 reports time slices in microseconds (millionths of a second).

The `-s` *config_file* option uses *config_file* to set scheduler parameters for the specified class. The configuration file must be in the class-specific format produced by the `-g` option. The meanings of the parameters are described in the sections above on the scheduler parameter tables; `ts_dptbl` holds time-sharing parameters and `rt_dptbl` holds real-time parameters.

The following examples show how to set the parameters for the default classes as specified in the configuration files `rt_config` and `ts_config`. The examples presuppose that these two files are in the correct formats.

```
$ dispadmin -c RT -s rt_config      # set real-time parameters

$ dispadmin -c TS -s ts_config      # set time-sharing parameters
```

The files that specify the new scheduler parameters must have the same number of priority levels as the current table that is being overwritten. To change the number of priority levels, you must change the `ts` file or the `rt` file in the `master.d` directory as described in the section above on configuration.

11 Restore Service

Introduction

This chapter tells you how to restore backed up files, directories, file systems, data partitions, disks, and partitioning information from archive volumes. You can do any of these functions by selecting the appropriate task from a series of menus provided for administration. To access the system administration menu for using the restore service, type

```
sysadm restore_service
```

The following menu will appear on your screen:

```
1         Restore Service Management

operator  - Set/Display the Restore Operator
respond   - Respond to Restore Job Prompts
restore   - Restore from Backup Archives
status    - Modify/Report Pending Restore Request Status
```

If you prefer not to use the menus, you can perform the same tasks by executing shell-level commands instead. The following table shows the shell commands that correspond to the tasks on the menu.

Task to Be Performed	*sysadm* Task	Shell Command
Restore from backup archives	`restore`	`restore`(1M) `urestore`(1M)
Respond to pending restore requests	`respond`	`rsoper`(1M)
Cancel unsatisfied requests, or remove satisfied restore requests	`status`	`rsoper`(1M)
Set up and/or display the login name of the operator who will monitor restore requests	`operator`	`rsnotify`(1M)
Obtain status information on pending restore requests	`status`	`rsstatus`(1M) `ursstatus`(1M)

Each task listed above is explained fully later in this chapter. In addition, the *System Administrator's Reference Manual* and *User's Reference Manual* provide information on the shell commands.

Some of the `sysadm` menus do not offer all the functionality available with the corresponding shell-level commands. If you are not an experienced administrator, however, you may find it easier to use the menus, which provide prompts and help messages that are not available with shell commands.

Discussion of System Restores

This chapter includes a section on performing full and partial system restores. These functions are separate from the restore service that is available to all users. They must be performed by the administrator in firmware mode. See the section entitled "System Restores."

Overview of Restore Operations

The restore service enables you to retrieve copies (archives) of files, directories, file systems, data partitions, disks, and partitioning information that have been preserved on archive media such as diskettes and tapes. There are various types of restore operations; the type you do depends on the type of archive you have. Therefore, to fully understand restore operations, you should familiarize yourself with the "Backup Service" chapter before reading this chapter.

On large computer systems, the tasks associated with the restore service are usually performed by two people: a system administrator and a computer operator. (On small systems, one person may serve as both the administrator and the operator.) The system administrator's job is

- to establish backup policies based on factors such as available resources, the needs of users, and management directives
- to decide which storage media are used for restore operations and how these media are organized and used
- to issue restore requests for entire file systems, data partitions, directories, and files

The computer operator's job is

- to check the status of pending restore requests
- to respond to system prompts when a restore request requires assistance
- to insert and remove storage media

Users on the system may not perform restore operations themselves. They may submit requests for restore operations to the operator and periodically check the status of operations in progress.

How Restore Requests Are Identified

The restore service assigns a job ID only if a restore request cannot be satisfied automatically and needs operator assistance. A job ID is an alphanumeric string consisting of the word `rest`, a dash, and a numeric ID. When a job ID is assigned to a restore request for multiple individual files, the restore service identifies each file in the request by appending a letter to the job ID for each file. This labeling convention is useful because it allows individual parts of a request to be handled separately. For example, a user might request a restore of the files `myfile.1`, `myfile.2`, and `myfile.3`. The job IDs for these files are `rest-4592a`, `rest-4592b`, and `rest-4592c`, respectively. The user decides that it is not necessary to restore `myfile.3` so he or she cancels the request for that file. The operator can then cancel `rest-4592c` without affecting the restore requests for the other two files.

Note that if these three files are contained in a directory, and if the directory is the object to be restored, then only one job ID is assigned to the directory restore no matter how many files are in that directory.

How Restore Operations Are Performed

The restore process begins when someone, either a user or an administrator, requests that a particular object be restored. This is done by issuing either the `restore` or `urestore` shell command or by using the `restore` menu item.

Once a restore request is issued, the restore service first looks at the backup history log to see if a backup has been performed for the object to be restored and determines if the archive is online.

If all this is true, the user receives the following immediate message:

```
Attempting restore from online archive
```

If the restore service cannot find an archive of that object for any reason, the user will receive a message similar to the following:

```
object:

urestore: Restore request id for object is rest-123a
```

This could happen if the archive is not online or if the restore service cannot find an archive with that information.

In the latter case the operator must then determine which archive might contain the object to be restored. Some restores might require multiple archives.

NOTE If the operator finds an archive produced on a pre-UNIX System V Release 4.0 system, it may or may not be possible to restore that object with the Release 4.0 restore service, depending on the command used for the backup. If the backup was done with the `cpio` command, the operator can restore the archive the same way he or she would restore any Release 4.0 archive. However, if the `volcopy` command was used for the backup, the item cannot be restored using the Release 4.0 restore service. Instead, it must be restored by using the menus for the pre-Release 4.0 restore service. (The pre-Release 4.0 restore service menu can be accessed by typing `sysadm old_sysadm` and then selecting the `filemgmt` item from the Main Menu.)

If the necessary archives are mounted, the restore operation proceeds automatically. Otherwise a `mail` message is sent to the operator indicating that a restore request is pending. This message reads as follows:

```
There is a pending restore request from user_login.
```

The operator can periodically look up pending restore requests that need servicing by issuing the `rsstatus` command or by using the `restore_status` menu item.

The last step of the restore process is to service pending restore requests. The operator loads the appropriate storage medium and then either issues the `rsoper` command or selects the `respond` item from the `restore_service` menu.

How the Restore Service Works

When the restore service tries to see if the archive is online, it does so by first looking to see if the object is listed in the history log. The restore service then tries to identify the exact archive that contains the object.

If the restore service is processing a request to restore a file, it looks through the archives available for all dates until just before the date from which the object is to be restored. The restore service chooses the most recent archive but not one that is more recent than the requested date.

On the other hand, if the restore service is processing a request to restore any object other than a file, it looks through the archives to find the most recent full file backup performed before the date from which the object is to be restored.

The restore service will restore that full file backup and update it with any incremental file backups performed up to the requested restore date.

As stated before, if the archive is found online, the restore proceeds automatically. Otherwise mail is sent to the operator and the request waits for service.

As explained in the "Backup Service" chapter, an archive is created by backing up requested objects using one of six default backup methods. The method chosen for the backup operation affects the type of restore that can be done and the level of difficulty of that restore operation. For example, data partitions can be restored only from full data partition backup archives.

Preparing for Restore Jobs

When an administrator is establishing restore policies for a computer site, one of the choices is who should be responsible for servicing restore requests. If the administrator decides to delegate this job to an operator, the `rsnotify` command can be used to route service requests to the person selected. For example, the command

```
rsnotify -u user
```

causes mail to be sent to *user* when a request is made. If `rsnotify` is invoked without options, the login of the assigned operator is displayed. The display also includes the date the operator was assigned. If `rsnotify -u` *user* is invoked when an operator has been assigned, the existing assignment is replaced by the new one.

Once an operator has been assigned, the operator receives mail about pending jobs and can scan for pending restore requests with the `rsstatus` command or the `status` item from the `restore_service` menu.

If no operator is assigned, `mail` messages about restore requests are sent to `root` instead.

Using the Restore Service

All users can request restores of any files and directories that they own, but only administrators can request restores of data partitions, file systems or entire disks. Because of this division of functionality there are two commands for requesting restores. The `urestore` command can be used by anyone to request a restore of one of their own files or directories. (An administrator can use this `urestore`—from the the root prompt—for any file or directory, regardless of who owns it.) The `restore` command can be used only by an administrator. `restore` is used to request a restore of file systems, disks or data partitions, or to display partitioning information.

By default the restore service retrieves the latest version of an object. However, any archived version can be restored by identifying the desired version on the command line.

Directory or File Restores

When issuing the `urestore` command, you must have read permission for the parent directories of the file or directory you are having restored. You must have write permission as well for the immediate parent directory. In addition, if the request is made by anyone other than the administrator or the operator, that person must have owned the file or directory at the time the archive was made. The following options are available with the `urestore` command:

`-c` *job_ID*	Cancels a previously issued restore request identified by *job_ID*.
`-d` *date*	Restores a file system or directory as of *date*, which may or may not be the date of the latest archive version. The value of *date* is the same ten-character string used to specify a month, day, hour, minute, and year with the `date(1)` command: *MMddhhmmyy*.
`-m`	If the restore cannot be carried out immediately, this option notifies the person requesting the restore (via `mail`) when the request has been completed.
`-n`	When issued with the `-D` or `-F` option (one of which must be used), the `-n` option displays a list of all archived versions of a file or directory contained in the backup history log. The `-n` option does not restore the file or directory.

`-o` *target*	Restores the archive to the *target* location.
`-D` *directory*	Restores *directory* and all the files underneath it.
`-F` *file_name*	Restores *file_name.*

Restoring Other Disk Objects

An administrator or operator can restore a data partition, a file system partition, or an entire disk by using the `restore` command, along with the following options:

`-A` *partdev*	Initiates a restore of the entire disk *partdev.*
`-P` *partdev*	Initiates a restore of the data partition *partdev.*
`-S` *oname*	Initiates a restore of the file system partition *oname.*
`-n`	When issued with the `-A`, `-P`, or `-S` option (one of which must be used), the `-n` option displays a list of all archived versions (of the disk object) contained in the backup history log. The `-n` option does not restore any disk objects.

partdev takes the form `/dev/rdsk/c?d?s?` (`c?t?d?s?` for SCSI devices). If `-A` is specified, the full disk method is used to repartition the disk. When *partdev* is used as an argument to `-A` for the 3B2 computer, the value of *partdev* is `c?d?s6`.

CAUTION Restoring a disk overwrites any data on the disk and resets all partitioning information for the disk to that of the archive.

Restoring the Backup and Restore Services

When restoring any file system containing the backup and restore services, you must restore the following components:

- the backup and restore shell level commands
- the backup methods found in the directory `/etc/bkup/method`
- the system-supplied backup and restore tables found in the directory `/etc/bkup` or any backup and restore tables you have defined

On most systems, these components reside in the root directory (/) and in the `/usr` file system, but they may be located in any file system.

The following is the procedure for restoring the backup and restore services:

1. Change the system state to single-user mode by typing `init S`.
2. Mount the diskettes containing the Essential Utilities. These utilities include the commands for the backup and restore services.
3. When the program asks if you want to reformat your disk, type `yes` (if you want default partitioning). You will need to have your reformatting specifications at hand so you can provide this information. (See "Full System Restore: Changing the Disk Partition Sizes.")
4. Restore the most recent copy of the backup history log for your system; this may be the system-supplied log or a log you have created. To restore the most recent copy of the system-supplied log, type

   ```
   urestore -F /etc/bkup/bkhist.tab /etc/device.tab \
   /etc/dgroup.tab
   ```

   To restore the most recent copy of your custom backup history log, type

   `urestore -F` *pathname*

   where *pathname* is the correct pathname to your backup history log.
5. To restore the `/usr` file system, type

   ```
   restore -S /usr
   ```

   Note that to do these functions, you must run `restore` rather than the `urestore` command.

6. Restore the lost file systems and data partitions. Use the backup history log display to determine the labels of archive volumes that contain the information to be restored.

Specific Archive Version Restores

You may want to restore a specific archive version of an object other than the most recent version, especially if the most recent version has been corrupted.

This is done by invoking `restore -n` with the `-A`, `-P`, or `-S` option. This command displays a list of all archived versions of the object by date. The `-n` option displays the version list only; it does not invoke the restore request. From the list, you can select the date of the correct version to be restored and then with the `restore` command include the `-d` *date* option, where the value of *date* is the same ten-character string used to specify a month, day, hour, minute, and year with the `date(1)` command: *MMddhhmmyy*.

Restoring an Object to a New Location

By default, the restore service restores an object to its original location. However, there may be times when you don't want to restore an object to its original location.

You can restore an object to a new location by running the `restore` command with the `-o` *target* option, where *target* is the name of the new destination location. Either command redirects the restore to a new location and thus avoids overwriting the contents of the previous location.

Checking the Status of Restore Requests

You can check the status of restore requests by using options to any of three commands: `restore`, `rsstatus`, or `ursstatus`.

The `-s` or `-v` options can be used with either the `restore` or `urestore` command. The `-s` option displays a "." for each 100 blocks transferred from the archive. The `-v` option displays the name of each directory or file as it is transferred from the archive. Note that these options take effect only when the required archive volume is on line and the restore operation is processing.

Furthermore, both the `rsstatus` and `ursstatus` commands provide a list of pending restore requests. The `rsstatus` command can be issued only by an administrator or operator. The `ursstatus` command can be used by anyone to view the status of any restore he or she has requested. `ursstatus` lists pending file and directory restore requests for the invoking user. `rsstatus` lists all the pending restore requests in detail.

Displaying the Status Table

The information gathered by either of these commands is recorded in the `/etc/bkup/rsstatus.tab` table. To display the contents of that table, invoke either command without options. An `rsstatus` report contains the following fields:

`Job_ID`	The job ID of the restore request
`Login`	The login name of the person requesting the restore
`File`	The full pathname of the file to be restored
`Date`	The specified date from which an object should be restored. (The restore service selects archives made on a backup date as near as possible to the date specified.)
`Target`	The full pathname of the location where the restored object should be placed
`Backup_Date`	The date of the last backup performed before the date specified in a restore request
`Method`	The method used to perform the backup
`Dtype`	The destination device, such as a diskette or cartridge tape, on which the backup archive was created
`Labels`	The labels for the archive volumes to be restored

The `ursstatus` report contains only the first five fields.

The `rsstatus` report appears in the following format on your screen:

```
Jobid        Login  File          Date    Target        Bkp date  Method   Dtype    Labels

rest-11111a  user1  /home/user1   Dec 27  /home/user1   Sun Dec   incfile  dlabl1   tlabl1
                    /oam/ISSUES   1989    /bkISSUES     27 1987
                                  17:00:
                                  00
rest-32984c  user1  /home/user1   Dec 27  /home/user1   Sun Dec   incfile  dlabl1   tlabl1
                    /oam/bkrs     1989    /oam/bkrs     27 1987
                                  17:00:
                                  00
rest-04818a  user2  /home/user2   Dec 27  /home/user2   Sun Dec   ffile    dlabl2   tlabl1
                    /rsstatus.c   1989    /rsstatus.c   27 1987
                                  17:00:
                                  00
rest-04818a  user2  /home/user2   Dec 27  /home/user2   Sun Dec   ffile    dlabl2   tlabl1
                    /myfile.c     1989    /myfile.c     27 1987
                                  17:00:
                                  21
rest-64978a  root   /usr          Dec 27  /dev/rd       Mon Jan   ffile    dlabl3,  tlabl3
                                  1989    sk/c1d0       18 1988                     dlabl4,
                                  17:00:  s2                                        dlabl5
                                  50
```

Customizing the Display of the Status Table

The last section described the restore status table and the default display of information in it. If you do not want to see a default report, however, you can customize both the contents and the format of the status report.

The default display provides all the available information about pending restore requests. You can restrict the fields displayed by using one of the following options to the `rsstatus` command: `-d` [*dtype*], `-j`, or `-u`.

`-d` [*dtype*]	Restricts the display to restore jobs that can be satisfied by a specific device. *dtype* specifies the device type (such as diskette or cartridge tape). The *dtype* argument is optional. When would an administrator want to use this argument? Specifying a device type may be helpful, for example, if an administrator's access to a particular device (such as a cartridge tape drive) is limited and

he or she therefore has little time in which to do restore operations on cartridge tape.

In this situation an administrator may need to ignore, temporarily, operations being done on devices that are available at any time, and concentrate on monitoring those operations that require a cartridge tape drive. By specifying `ctape` as an argument to the `-d` option, the administrator can display status reports for only those operations being done on cartridge tape.

`-d diskette`
: Displays the status of all restore requests that can be satisfied by inserting diskettes into a diskette drive.

In another instance, an operator may have mounted a certain set of archive volumes and might want to verify that all restores that can be satisfied by those volumes will be performed.

`-d:bk0010,bk0011,bk0012`
: Displays the status of all requests for restore operations from an archive on the volumes with the labels specified (`bk0010`, `bk0011`, and `bk0012`).

`-j` *job_IDs* Displays the status of operations with the specified restore job IDs

`-u` *users* Displays the status of restore operations requested by the specified user logins

You can specify all three options (`-d`, `-j`, and `-u`) on the same command line. When you do, the report displayed will contain only those entries that satisfy all three specifications.

In the default `rsstatus` display, each field has a title and a specific length. Data entries that exceed the specified field length wrap to the next line within the field. You can change the format of the `rsstatus` display by using the `-h`, `-f`, or `-s` option to the `rsstatus` command.

`-h` Suppresses the headers (titles) for a display. This option is useful when the contents of a display are to be filtered by another process.

`-s` Suppresses field wrap on the display so the data in the display appear in a single line. This option is often used with the `-f` option.

`-f` *field_separator*
Specifies a field separator to be used when field wrap is suppressed. The value of *field_separator* is the character that will appear as the field separator in the display. For clarity, do not choose a character that is likely to appear in a field. For example, do not use a colon as a field separator if the display will contain dates in which a colon is used to separate hours from minutes.

Servicing Pending Restore Requests

Restore operations that cannot be performed immediately are posted to the restore status table and are considered "pending." Pending restore jobs must be serviced by an operator who received mail about the pending request and services it through the `rsoper` command. An operator can usually satisfy a pending restore request by locating and installing the correct archive volume.

You can service a pending restore job, by completing the following procedure.

1. Display the restore status table by running `rsstatus` with the desired options. Note the name of the object to be restored and the label of the archive volume for it.

   NOTE If there is no label information in the status table, you may have to determine which archive volume contains the backup from which to restore by guessing and relying on other information. For example, the date of a backup may allow you to identify an archive volume.

2. Mount the archive volume.

3. Invoke `rsoper -d` *ddev* where *ddev* is the name of the device that is to read the archive. *ddev* takes the following form:

   ddevice[*:dchar*][*:dlabels*]

dlabel must be specified if there is more than one label. It must include a device name and it may include device characteristics and volume labels.

NOTE If the history log has been removed, all of the *ddev* fields (*ddevice, dchar, dlabels*) must be specified.

4. After the volume is processed, check the restore status with the `rsstatus` command.

The restore service compares the information you have entered on the `rsoper` command line with the information in the restore status table and on the archive volume. If the information on the command line matches the information in the restore status table, the restore operation begins. If the information on the command line does not match that in the restore status table, the information on the command line predominates and the restore operation begins. Then the restore service attempts to resolve any restore requests that can be satisfied by the archive volume described by `-d` *ddev*.

Options to `rsoper`

There are several options that can be specified with `rsoper`. These options are used only if you need to override the information in the restore status table. Archives made with the UNIX System V Release 4.0 backup methods contain the information needed for all restore operations except those performed with the `-d` option. Thus, the following options will probably be needed only for archives made on systems running pre-Release 4.0 releases of the UNIX system. (These archives may not contain all the information required by the Release 4.0 restore service.)

Basic Options

Three options allow you to describe the archive volume being mounted for use by a restore operation: `-t`, `-o`, and `-m`.

`-t`	Tells the service that the volume inserted in the destination device contains a table of contents for the archive.

−o *oname:odev*
: Specifies the originating file system partition or data partition to be restored. *oname* is the name of the originating file system; the value of *oname* may be null. *odev* is the device name of the originating file system or data partition.

−m *method*
: Specifies that the first archive volume in the destination device was created by the backup *method* specified.

The following example illustrates the use of some of these options. Suppose the backup history log no longer contains a log of the backup operations for which archives are being requested. To obtain the desired data, you must request an incremental file restore from the `/usr2` file system. To request this, enter

```
rsoper -d /dev/diskette2::arc.dec79.a,arc.dec79.b,x \
   arc.dec79.c -m incfile -o /usr2
```

The `/usr2` file system archive is found on diskettes labeled `arc.dec79.a`, `arc.dec79.b`, and `arc.dec79.c`.

Restricting Restores

By default, when `rsoper` is invoked, the restore service attempts to complete any restore requests on the archive volume mounted. However, you can restrict restore jobs to those with specific job-IDs by using the `-j` option. You can also restrict restore jobs to specific user logins through the −u option.

Removing and Canceling Restore Jobs

Entries in the restore status table may be deleted for either of two reasons:

- to remove a restore request that has been satisfied
- to cancel a job that cannot be completed or cannot be serviced at all

To remove an entry for a pending restore request, type

```
rsoper -r job_IDs
```

where *job_IDs* is a list of pending restore requests that have been serviced. This command notifies the people who issued these requests that the restore operations have been done successfully and that the entries for them have been removed from the status table.

To cancel a request, type

```
        rsoper -c job_IDs
```

where *job_IDs* is a list of pending restore requests to be canceled. This command notifies the people who issued these requests that the requests cannot be serviced and have been canceled.

System Restores

Full and partial system restores are altogether different from other types of restore operations. System restore operations are not done by issuing the `restore` command or any related commands; rather, they are done by rebooting the Essential Utilities diskette.

Partial system restores are required when a portion of the system has become corrupted or the administrator has forgotten the root password. Full system restores are necessary when there is a new system or disk, or when you need to increase the size of the core file system data partition.

- A partial system restore replaces (overwrites) the core system files on a hard disk with those originally distributed (on the core diskettes). These files include the Essential Utilities; user files are not affected by a partial system restore.
- A full system restore erases everything on a hard disk and then loads the core system files.

To request either a full or partial system restore, you must be in single-user state, and you must mount `/usr`.

NOTE A full system restore erases everything on a disk. If you use this procedure to change partition sizes, do a complete file disk backup first.

Partial System Restore

This type of restore is useful if you have forgotten your root password or if your system has been corrupted.

During a partial system restore, certain system files are overwritten, including the terminal configuration and password files. (These files are needed to rebuild your previous system configuration.)

Use the following procedure to do a partial system restore on a 3B2 computer:

1. Change directory to the root directory.

2. Change the system state to firmware mode (system state 5) by typing `init 5`.
3. Insert the first Essential Utilities diskette into the diskette drive.
4. Boot the operating system by selecting option `0` from the instructions displayed on your screen.
5. Initiate a partial system restore by choosing option 2 from the instructions displayed on your screen.
6. Follow the instructions on the screen to remove and insert the Essential Utilities diskettes. When the last diskette has been copied, the system will reboot from the hard disk.
7. When the system is ready, you must rebuild the system files. You can do this in either of two ways: (1) you can follow the displayed instructions to access the `sysadm syssetup` menu; or (2) you can use your own procedure (see Step 8).
8. Recover, from backup copies, any system files not automatically saved, and reconfigure the system as necessary. System files automatically saved during this procedure are found in `/var/old` and include the following:

```
/bin/ed                     /etc/passwd
/bin/red                    /etc/profile
/dgn/.edt_swapp             /etc/rc2.d/S18setuname
/dgn/edt_data               /etc/rstrat.dat
/etc/TIMEZONE               /etc/saf/token/_config
/etc/bkup/bkexcept.tab      /etc/saf/token/_pmtab
/etc/bkup/bkhist.tab        /etc/saf/_sactab
/etc/bkup/bkreg.tab         /etc/saf/_sysconfig
/etc/bkup/rsnotify.tab      /etc/shadow
/etc/checklist              /etc/shutdown.d/*
/etc/cron/.proto            /etc/system
/etc/cron/at.allow          /etc/ttydefs
/etc/cron/cron.allow        /etc/vfstab
/etc/cron/queuedefs         /usr/lib/uucp/*
/etc/defaults/*             /usr/sadm/sysadm/menu/main.menu
/etc/device.tab             /var/sadm/bkup/*/bkexcept.tab
/etc/dgroup.tab             /var/sadm/bkup/*/bkreg.tab
/etc/disk.tab               /var/sadm/install/admin/default
/etc/group                  /var/sadm/install/contents
/etc/init.d/*               /var/spool/cron/crontabs/adm
/etc/inittab                /var/spool/cron/crontabs/root
/etc/master.d/*             /var/spool/cron/crontabs/sys
/etc/motd                   /var/spool/cron/crontabs/sysadm
```

If you are doing this restore operation because of a forgotten root password, copy all the above files back into their proper places and then add the new root password by typing

```
passwd root
```

If you are doing this restore operation because of system corruption, examine each of the files listed above carefully before putting them back in your system. One of these files may have caused the corruption.

After this procedure has been completed, all software packages should be reinstalled to insure that the system configuration is properly restored.

Full System Restore: Using the Default Disk Partition Size

A full system restore erases everything on a hard disk and then loads the core system files.

NOTE Because all files on hard disk are destroyed during a full system restore, a full system restore should not be performed until you have backed up (onto floppy diskette or tape) all files you want to keep.

This procedure allows you to perform a full system restore while maintaining the default disk partition sizes. Skip to the next section if you want to perform a full system restore that changes the disk partition size.

The following is the procedure for doing a full system restore (using the default disk partition size) on a 3B2 computer:

1. Change directory to the root directory.
2. Change the system state to firmware mode (system state 5) by typing `init 5`.
3. Boot the operating system by selecting option `0` from the instructions displayed on your screen.
4. Initiate a full system restore by choosing option `1` from the instructions displayed on your screen.
5. Type `y` after the following prompt, as shown:

   ```
   Use the default hard disk partitioning?
      [y, n, help]: y
   ```

6. Follow the instructions displayed on the screen to remove and insert the Essential Utilities diskettes. When the last diskette has been copied, the system will reboot from the hard disk.
7. When the system is ready, you must rebuild the system files. You can do this in either of two ways: (1) you can follow the displayed instructions to access the `sysadm syssetup` menu; or (2) you can use your own procedure.

Full System Restore: Changing the Disk Partition Sizes

This procedure differs from the one above because it allows you to change the disk partition sizes.

Before you begin, make sure you have calculated the number of blocks to be allocated to the various partitions, find out how much space is left on each disk, and determine the current disk partition sizes.

To determine how much space is available on a disk, run the `df -t` command; to determine the current disk partition size, issue the `prtvtoc` command and specify each disk for which you want to know the size. (Sizes reported by the `prtvtoc` command are not exact; this command rounds up all disk partition sizes reported.)

Enter these commands as follows:

```
df -t
prtvtoc /dev/rdsk/c1d0s6
prtvtoc /dev/rdsk/c1d1s6
```

Record the results.

To do a full system restore (changing the disk partition size) on a 3B2 computer, complete the following procedure:

1. Change directory to the root directory.
2. Change the system state to firmware mode (system state 5) by typing `init 5`.
3. Boot the operating system by selecting option `0` from the instructions displayed on your screen.
4. Initiate a full system restore by choosing option `1` from the instructions displayed on your screen.
5. Type `n` at the following prompt as shown:

```
Use the default hard disk partitioning?
   [y, n, help]: n
```

6. The next prompt is

   ```
   How many blocks for the swap partition?
      [ (range 3500 through max) quit help ](default n):
   ```

   Enter the number of blocks you want to allocate (*max* will vary according to disk size and *n* will depend on how your disk is partitioned). To determine how many blocks to allocate, use the figures you worked out beforehand. Remember, if you increase the default number of blocks in one partition, there will be fewer blocks available for the remaining partitions.

7. The system then prompts you for similar information about the next partition on the disk. Continue supplying information until you have partitioned all the blocks.

8. When the system is ready, you must rebuild the system files. You can do this in either of two ways: (1) you can follow the instructions displayed on the screen to access the `sysadm syssetup` menu; or (2) you can use your own procedure (see Step 9).

9. Recover, from backup copies, any system files not automatically saved, and reconfigure the system as necessary. The system files that are automatically saved (in `/var/old`) during this procedure include the following:

```
/bin/ed                         /etc/profile
/bin/red                        /etc/rc2.d/S18setuname
/dgn/.edt_swapp                 /etc/rstrat.dat
/dgn/edt_data                   /etc/saf/token/_config
/etc/TIMEZONE                   /etc/saf/token/_pmtab
/etc/bkup/bkexcept.tab          /etc/saf/_sactab
/etc/bkup/bkhist.tab            /etc/saf/_sysconfig
/etc/bkup/bkreg.tab             /etc/shadow
/etc/bkup/rsnotify.tab          /etc/shutdown.d/*
/etc/checklist                  /etc/system
/etc/cron/.proto                /etc/ttydefs
/etc/cron/at.allow              /etc/vfstab
/etc/cron/cron.allow            /usr/lib/uucp/*
/etc/cron/queuedefs             /usr/sadm/sysadm/menu/main.menu
/etc/defaults/*                 /var/sadm/bkup/*/bkexcept.tab
/etc/device.tab                 /var/sadm/bkup/*/bkreg.tab
/etc/dgroup.tab                 /var/sadm/install/admin/default
/etc/disk.tab                   /var/sadm/install/contents
/etc/group                      /var/spool/cron/crontabs/adm
/etc/init.d/*                   /var/spool/cron/crontabs/root
/etc/inittab                    /var/spool/cron/crontabs/sys
/etc/master.d/*                 /var/spool/cron/crontabs/sysadm
/etc/motd
/etc/passwd
```

Quick Reference to the Restore Service

The Administrator's Tasks

- Assigning an operator to service restore operations:

  `rsnotify -u` *user*

  where *user* is the login name of the operator.

- Displaying the name of the operator assigned to service restore operations:

  `rsnotify`

- Initiating a restore of a data partition:

  `restore -P` *partdev*

  where *partdev* is the name of a data partition.

- Initiating a restore of a file system partition:

  `restore -S` *oname*

  where *odevice* is the name of the file system partition to be restored.

- Initiating a restore of an entire disk:

  `restore -A` *partdev*

  where *partdev* is the name of the disk to be restored.

- Restoring an object from archive volumes of a particular date:

  `restore -d` *date* or `urestore -d` *date*

  where *date* is the date of the archive to be used for the restore. The value of *date* is the same ten-character string used to specify a month, day, hour, minute, and year with the `date(1)` command: *MMddhhmmyy*.

- Restoring an object to a new disk location:

  `restore -o` *target* or `urestore -o` *target*

  where *target* is the complete pathname of the destination location on the disk.

The Operator's Tasks

- Displaying a list of all archived versions of an object:

  `restore -n` or `urestore -n`

- Displaying the complete contents of the restore status table:

  `rsstatus`

- Displaying a list of restore jobs that could be satisfied by a specified device type or archive volume:

  `rsstatus -d` [*dtype*][:*dlabels*]

  where *dtype* is a description of a device type (diskette, ctape, and so on) and *dlabels* is the label of a particular archive volume (such as `bk0010`, `bk0011`).

- Displaying the status of particular `fdisk`, `fimage`, `ffile`, or `fdp` restore jobs:

  `rsstatus -j` *job_IDs*

  where *job_IDs* is a list of one or more job IDs for requested restores.

- Displaying the status of restore requests made by specific users:

  `rsstatus -u` *users*

  where *users* is a list of one or more user login names.

- Removing a pending restore request from the restore status table and designating it canceled:

  `rsoper -c` *job_IDs*

  where *job_IDs* is a list of one or more job IDs for requested restores.

- Removing a pending restore request from the restore status table and designating it complete:

  `rsoper -r` *job_IDs*

  where *job_IDs* is a list of one or more job IDs for requested restores.

- Servicing a pending restore request:

  `rsoper -d` *ddev*

  where *ddev* describes the device to be used to read the archive containing the file system or data partition to be restored.

- Suppressing field wrap and specifying an output field separator on the restore status display:

  `rsstatus -f` *c*

  where *c* is the character that will appear as the field separator in the display of the restore status table.

- Suppressing headers on the restore status display:

  ```
  rsstatus -h
  ```

Non-Privileged Tasks

- Canceling a previously requested restore:

  `urestore -c` *job_ID*

  where *job_ID* is the job ID of a job to be canceled.

- Displaying the status of particular `incfile` restore jobs:

  `ursstatus -j` *job_IDs*

  where *job_IDs* is a list of one or more job IDs for requested restores.

- Initiating a restore of a directory:

  `urestore -D` *directory*

  where *directory* is the name of the directory to be restored.

- Initiating a restore of a file:

  `urestore -F` *filename*

  where *filename* is the name of the file to be restored.

12 Security

Introduction

The UNIX operating system provides extensive features to maintain system security. However, no computer system is secure unless good standards of administration and use are established and followed. This chapter provides details about security for UNIX System V Release 4.0 and describes computer center practices that can enhance the security of your computer operations.

There is no individual administration menu dedicated to system security. However, there are several tools to help maintain security that are described in this chapter. Some of these are available through the `systemsetup` and `users` menus under `sysadm`(1M). If you prefer to work at the shell level you can do so instead of using the `sysadm` interface. The following table shows the shell commands that correspond to the tasks in the `sysadm` interface. Not all security-related commands discussed in this chapter have corresponding `sysadm` menu items.

Figure 12-1: Menus and Shell Commands for Performing Some Security Related Tasks

Action to be Performed	sysadm Task*	Shell Command
Display password aging for a user	`users/password`	`passwd -s` *user*
Change password for user	`users/password`	`passwd` *user*
Turn on aging, set *max*	`users/password`	`passwd -x` *max user*
Turn on aging, set *min*	`users/password`	`passwd -n` *min user*
Turn off aging	`users/password`	`passwd -x -1` *user*
Lock a user's login	`users/password`	`passwd -l` *user*
Set password for administrative or system logins	`systemsetup/password`	`passwd` *admlogin-ID*

* Many of these actions may be done via the commands available for the administration of users. See **sysadm**(1M) in the *System Administrator's Reference Manual* for all available options and an explanation of their capabilities.

Each task listed above is explained later in this chapter. In addition, the *System Administrator's Reference Manual*, and *User's Reference Manual* provide manual pages for these shell commands.

Overview of Security Administration

Security is an aspect of the operation of your computer that must always be kept in mind. A machine connected to phone lines or a local network has the potential for intruders. Even an isolated machine is subject to idle browsing by its legitimate users. Consider the possible loss if a file is altered or destroyed, or if the wrong person sees it.

This chapter covers:

- Important security concepts and guidelines for the UNIX system—how to control user and group access to directories and files
- Logins and passwords—what precautions to take when changing a password (some of which may be required by company policies) and how to assign or change a password
- Assigning special administrative passwords—what are administrative logins and why they should be protected
- Logging login attempts—how to start a procedure to track unsuccessful attempts to log in to the system
- Set-UID and set-GID—how to determine if unauthorized programs conditioned to execute via an administrative login exist
- A quick reference section that lists the security-related commands discussed in this chapter

Suggestions for Making Your System Secure

The security of any system is ultimately the responsibility of all who have access to it. As the administrator of your system, you need to consider the following:

- Restrict physical access to your computer (especially if it is a small machine) so that someone does not simply walk off with it.
- Set the access permissions to directories and files so that they can be accessed only as needed by the owner, group, or others. Publicly writable directories are a security hazard. Allow them only if you have a good reason.
- Assign passwords to all logins and change them regularly. You can force them to be changed regularly by implementing password aging. Do not pick obvious passwords: six-to-eight character nonsense strings using letters and numbers are recommended over recognizable words. Remove or lock logins that are no longer needed.
- Do not keep sensitive information on a system with dial-up ports; the security of any system with dial-up ports is difficult to guarantee.
- Users who make use of the su command to become `root`, or any other user, can compromise the security of your system by accessing files belonging to other users without the other users' knowledge. For this reason, a log is kept on the use of the su(1) command. Check the file `/var/adm/sulog` to monitor use of this command. The format of `/var/adm/sulog` is described in Appendix B, "Directories and Files."
- Keep in mind that login directories, user profile files, and files in `/sbin`, `/usr/sbin`, and `/etc` that are writable by others are security give-aways.
- Encrypt sensitive data files. The `crypt`(1) command together with the encryption capabilities of the editors (ed and `vi`) provide better protection for sensitive information. The Security Administration Utilities package (domestic customers only) must be installed before you can run `crypt`(1).
- Do not leave a logged-in terminal unattended, especially if you are logged in as `root`. If you must be away from your terminal, log off before leaving.
- Place an appropriate umask command in the system profile (`/etc/profile`) to set a default security level for file creation.

- As system administrator, use full pathnames for critical commands (for example `/usr/bin/su` instead of `su`).
- Don't mount a medium (such as a floppy disk) unless the contents are trusted. These file systems may contain set-user-ID or Trojan horse (undesirable gift) programs.
- Don't add packages or programs from untrusted sources. This is the most common way of spreading computer viruses.
- For more information on network security and dialup passwords, see the "Network Services" chapter.

Logins and Passwords

To log in to the UNIX system, a user must enter both a login name and a password. Although logins are publicly known, passwords must be kept secret, known only to their owners. To enhance the security of your system and data, we recommend that you ask your users to change their passwords occasionally. For a high level of security, normal users should do so about every 6 weeks. System administration logins (such as root and sys) should be changed monthly or whenever a person having the root password leaves the company or is reassigned. Although voluntary compliance with this practice is desired, the UNIX operating system provides a mechanism to force compliance. This mechanism is called password aging.

Choosing a Password

Most security breakins of computer systems involve guessing the person's password. While the passwd(1) command has some criteria for making sure the password is hard to obtain using mechanical means, a clever person can sometimes guess a password just by knowing something about the person and habits.

- Bad choices: names of family members or pets, car license numbers or telephone numbers, Social Security number, employee number, names related to a person's hobbies or interest, words currently popular in the media (such as slang from TV shows), seasonal themes (such as "turkey" in November or "superbowl" in January). Also, any variations on this by substitution or addition of a special character.
- Good choices: puns, words in a foreign language, a word reversed (yekrut for turkey), or a nonsense word made up of the first letter of every word in a phrase (Mhallifwwas – Mary had a little lamb, its fleece was white as snow).
- Add a non-alphabetic character in the middle of the password (be careful about magic characters such as # and @ and control characters). Substitute a number for a similar letter (for example 0 for o, 3 for e, 1 for l or i).
- Remember that the clever person is aware of the above as well.

Password Aging

The password aging mechanism forces users to change their passwords on a periodic basis. Provisions are made to prevent a user from changing a new password before a specified interval. Password aging is selectively applied to logins by using the passwd(1) command. If you require more access control than what is provided by password aging, you can also change /etc/profile to require a second access code as part of the login process (see the "User and Group Management" chapter).

The password aging information requires setting the following parameters for each login:

min	the minimum number of days required between password changes
max	the maximum number of days the password is valid
warn	the number of days before the password must change that a warning message will begin appearing to the user

As a result of using passwd(1), the following parameter will also have changed:

lastchanged	the number of days between January 1, 1970, and the date that the password was last modified

Displaying Password Information

Password and aging information can be displayed using the -s option of the passwd command. For example, if you type:

```
passwd -s sms
```

the following information will appear if there is password aging.

```
sms  PS  06/23/90  14  84  7
```

If password aging is not turned on, only the first two fields will appear. The six fields contain the following information:

- login name (sms)

- password status (PS)
  The following strings may appear:

Status Type	Symbol
No password for this login	NP
Login is locked	LK
Anything else	PS

- date the password was last changed (06/23/90)
- minimum number of days after *lastchanged* before the user can change the password (14)
- maximum number of days after *lastchanged* until the user will be forced to change the password (84)
- number of warning days before the password must be changed (7)

Thus, the information obtained for this example shows that there is a password for the login sms that cannot be changed before July 6 and that must be changed by September 15, 1990. On September 8, 1990, this user will begin seeing a warning message that the password will expire and should be changed.

To display the password status and aging information for all users on your system, use the -a option to the passwd command, instead of specifying individual logins:

```
passwd -s -a
```

Only a privileged user can use the -a option for the passwd command.

Sample passwd Commands

Password administration can be set up in a variety of ways to meet the needs of different organizations. Some examples are discussed in the following sections.

1. Change someone's password

   passwd *login_name*

   Because this command is run by the administrator, no prompt for the old password is given. Instead, as a privileged user, the administrator is prompted to enter the new password. The password is not displayed as it is typed. The command requires you to enter the password twice to assure it is typed accurately.

2. Turn on aging, set *max* to 84 and *min* to 7 days, respectively.

   ```
   passwd -x 84 -n 7 login_name
   ```

3. Force a user to change the password at the next login session.

   ```
   passwd -f login_name
   ```

4. Lock a password, set *max* to 7 and *min* to 10 days.

   ```
   passwd -x 7 -n 10 login_name
   ```

   Because *min* is greater than *max*, the password is locked and cannot be changed but the user can still log into the system. Only root can change this password.

5. Turn off aging by setting *max* to negative one.

   ```
   passwd -x -1 login_name
   ```

6. Warn the user starting 14 days before the password is set to expire that a new password must be chosen.

   ```
   passwd -w 14 login_name
   ```

   Starting 14 days before *max*, the user will see the message:

   ```
   Your password will expire in 14 days
   ```

   Each day, the number will decrease until the password expires or the user changes the password.

For more information, see passwd(1) in the *User's Reference Manual*.

Dial-up Passwords

A dial-up password is an additional password you must enter before you are allowed access to the system. This built-in UNIX system capability can be added to enhance the system's security.

A dial-up password can be changed only by the system administrator, who, to ensure the integrity of the system, should change the password about once a month, usually on the first of the month.

Creating a Dial-up Password

When you first establish a dial-up password, make certain to remain logged in on at least one terminal while testing the password on a different terminal. If you make a mistake while installing the extra password and log off to test the new password, you might not be able to log back on. If you are still logged in on another terminal, you can go back and fix your mistake.

To institute dial-up password protection on your system, you need to create two files:

- `/etc/dialups`, containing a list of the terminal devices on which dial-up password will be required, and
- `/etc/d_passwd`, containing the encrypted password and the login programs that require the user to enter a password before they can be invoked.

The modes of the files should be `600`, and they should have owner and group set to `root`.

The `/etc/dialups` file is a list of terminal devices. It should look similar to this:

```
/dev/term/21
/dev/term/22
/dev/term/23
```

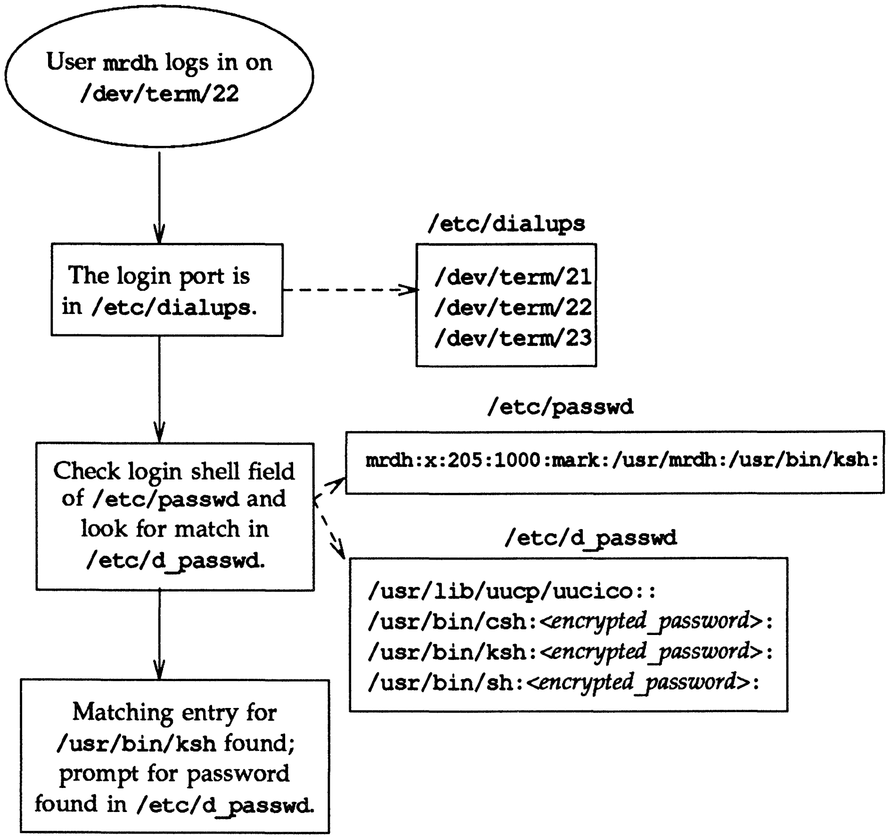
It lists all ports that require the extra security provided by a dial-up password. These are actually modem ports on the system.

The /etc/d_passwd file looks similar to this:

```
/usr/lib/uucp/uucico::
/usr/bin/csh:encrypted_password:
/usr/bin/ksh:encrypted_password:
/usr/bin/sh:encrypted_password:
```

When a user attempts to log in on any of the ports listed in /etc/dialups, the login program looks at /etc/d_passwd and may prompt the user for a second password. Whether a second password is asked for or not depends upon the login shell that is specified in the shell field of the user's login entry in the /etc/passwd file, and whether or not this login shell has an entry in /etc/d_passwd. The basic sequence is best illustrated by the following figure.

Figure 12-2: Basic Dialup Password Sequence


User mrdh logs in on /dev/term/22
The login port is in /etc/dialups.
/etc/dialups
/dev/term/21
/dev/term/22
/dev/term/23
Check login shell field of /etc/passwd and look for match in /etc/d_passwd.
/etc/passwd
mrdh:x:205:1000:mark:/usr/mrdh:/usr/bin/ksh:
/etc/d_passwd
/usr/lib/uucp/uucico::
/usr/bin/csh:<encrypted_password>:
/usr/bin/ksh:<encrypted_password>:
/usr/bin/sh:<encrypted_password>:
Matching entry for /usr/bin/ksh found; prompt for password found in /etc/d_passwd.

Because most users will be running a shell when they log in, all shell programs should have entries in /etc/d_passwd. Such programs include uucico, sh, ksh, and csh. If some users run something else as their login shell, include it in the file, too.

Each entry in /etc/d_passwd has two fields, demarcated by semicolons. The first field is the login program that will require a dial-up password. The second field contains the encrypted password and will be discussed later.

In our example above, the uucico program does not have anything in this field. This allows remote systems to call your system, via uucp, without having to know the dial-up password. The uucp subsystem is relatively secure, if properly administered, and usually does not need a dial-up password. However, if you are concerned about security here, it would be wise to require a password for uucp, too. The dial-up password need not be the same for uucp as for ksh, sh, etc.

The entry for /usr/bin/sh defines the default dial-up password. If the user's login program is not found in /etc/d_passwd, or if the login shell field in /etc/passwd is null, this password entry will be used.

If there is no entry for /usr/bin/sh, users whose shell field in /etc/passwd is null or does not match any entry in /etc/d_passwd will not be prompted for a dial-up password.

Note that the /etc/d_passwd file could be used to temporarily disable dial-up logins by putting an entry such as:

```
/usr/bin/sh:*:
```

by itself in the file.

A dial-up password can be created by following these steps

1. Using useradd or sysadm's "add user" menu, add a "dummy" user, say dummy.
2. Give it a password with the passwd(1) command or the sysadm form.
3. Capture the encrypted password from /etc/shadow by typing

   ```
   grep dummy /etc/shadow > dummy.temp
   ```

4. Using userdel or the appropriate sysadm form, delete the dummy user.
5. Edit dummy.temp and delete all fields except the encrypted password. Fields are delimited with a colon (:) and the password is the second field.

6. Edit the `/etc/d_passwd` file and read the encrypted password from your `dummy.temp` file in as the password field.

You should follow these steps every time you change the dial-up password.

Locking Unused Logins

If a login is not used or needed, you should do one of two things:

- use `userdel`(1M) to delete the login (see the "User and Group Management" chapter)
- disable (lock) the login

A login is locked by running the `passwd` command with the `-l` option.

```
passwd -l login_name
```

The string `LK` is displayed in the password field when the `passwd -s` command is issued. This shows that the login is locked; the user will not be allowed to log in. To unlock this login, the administrator must run `passwd` for the user.

Login Authorization

UNIX System V Release 4 provides two other login control mechanisms: login deactivation and expiration.

Login Expiration

As an administrator, you may want to create a login that can only be used for a short time. After this time, this login would "expire" and the owner would no longer be able to use it to log in without the administrator's intervention. To set the expiration date, type

```
useradd -e mm/dd/yy login_name
```

where *mm/dd/yy* is an absolute date:

mm	is a one or two-digit number representing the month (1-12)
dd	is a one or two-digit number representing the day of the month (1-31)
yy	is a two-digit number representing the year (00-99)

Setting or changing the expiration date may also be done with usermod(1M).

If a login's expiration date must be extended, you can do so by running usermod again with a new date as the argument to -e.

Deactivating a Login

It may be useful to know that a user has not logged into the system for a while. By setting the "inactive" field, a login will be considered inactive if a user has not logged in for a set number of days. If the user does not log in for this number of days, the login becomes inactive and the user will be prevented from logging in until the administrator resets the login. To set an inactive field, type:

```
usermod -f n login_name
```

where *n* is the number of days after *lastlogin* after which the login will be considered inactive.

Setting the inactive field may also be done when adding a new user to your system by using useradd(1M).

If a login has become inactive, it can be reactivated by completing the following steps:

1. Check the current inactive value

   ```
   logins -a -l login_name
   ```

   The first number on the second line of the output of this command is the inactive field.

2. Set the inactive field to zero.

   ```
   usermod -f 0 login_name
   ```

3. Have the affected user log in again so that the *lastlogin* date will be updated.

4. Reset the inactive field once again.

   ```
   usermod -f n login_name
   ```

   You can use the value found in Step 1 for *n*.

Displaying Login Information

The amount of warning time, number of days before inactivation and the expiration date can be displayed using the `logins` command. This command displays a variety of information about the users on your system. Of interest here are the -a and -x options. For information on other options and uses of this command, see the "User and Group Management" chapter and the `logins`(1M) command in the *System Administrator's Reference Manual*.

Login Logging

A logging mechanism exists that logs unsuccessful attempts to access your UNIX system. After a person makes five consecutive unsuccessful attempts to log in, all these attempts are logged in the file /var/adm/loginlog.

loginlog

To turn on the mechanism that logs unsuccessful attempts to access the system, the administrator must create the file /var/adm/loginlog. If this file exists and five consecutive unsuccessful login attempts occur, all are logged in loginlog(4) and then login sleeps for 20 seconds before dropping the line. If a person makes fewer than five unsuccessful attempts, none of them are logged.

If loginlog does not exist, five failed login attempts will still cause the system to sleep for 20 seconds and drop the line, but nothing will be logged.

The default status is for this text file not to exist and for logging to be off. To enable logging, create the log file with read and write permission for root only.

1. Reset the default file creation privileges in a separate shell.

   ```
   /bin/sh
   umask 066
   ```

2. Create the loginlog file.

   ```
   > /var/adm/loginlog
   ```

3. Set the group to sys.

   ```
   chgrp sys /var/adm/loginlog
   ```

4. Change the ownership of the file to root.

   ```
   chown root /var/adm/loginlog
   ```

5. Return from the newly created shell level.

   ```
   exit
   ```

This file may grow in size quickly. To use this information and to prevent the file from getting too large, it is important to check and to clear the contents of the loginlog file occasionally. A large number of lines in a short amount of time in this file may suggest an attempt to break into the system. For more information about this file, see loginlog(4) in the *System Administrator's Reference Manual*.

Last Login Time

When a user logs into the system, the time the login was last used will be displayed. We recommend that users check this time to make sure that it corresponds to the time they actually did log in. If it does not, an unauthorized use of that user's login may have occurred.

Recording su Use

One way to record all use of the su command is to print a message on the system console each time the command is run. To do this, add the line

```
        CONSOLE=/dev/console
```

to /etc/default/su.

Special Administrative and System Logins

There are two familiar ways to access the system: via either a conventional user login or the `root` login. If these were the only two ways to access the system, however, effective use of the system would have to be curtailed (because `root` would own many directories) or many users would have to know the `root` password (a bad security risk) or the system would be wide open (because `root` would own few directories). All these conditions are undesirable.

The solution to a good mix of system use and system security is available to you with the use of special system logins and administrative commands that can be password-protected (see the "System Setup" chapter for information on doing this). There are two types of special logins:

administrative	These commands, which are also logins, perform functions that might be needed by the users on your computer.
system	These logins allow privileges to be split into smaller domains so that fewer people have access to the entire system.

We recommend that all the following logins be password protected.

Figure 12-3: Administrative Logins and Uses

Function	UID	Use
`setup`	0	Set up the computer. Once the machine is set up, you do not want anyone doing it again without your knowledge.
`sysadm`	0	Allows access to administrative functions that do not require a user to log in as `root`.
`powerdown`	0	Power the computer down.
`checkfsys`	0	Begin a file system check on the specified file systems.
`makefsys`	0	Make a new file system on the specified media.
`mountfsys`	0	Mount the specified file system for use.
`umountfsys`	0	Unmount the specified, previously mounted file system.

The commands above allow access to selected directories and system functions. They may be used as login names at the `login` prompt as well as commands. If you log in to the system with one of these names, the system executes the command after login and exits to the `login` prompt once you quit or complete the function performed by the command.

Most of these administrative functions allow a user access to critical portions of the operating system. Therefore, it is recommended that you assign passwords to the commands above. Once you assign passwords to these commands, any user attempting to log on to your computer using one of these commands as a login (and any user attempting to execute one of these commands from the shell) is prompted for the password.

It is recommended that the passwords to the following system logins be distributed on a need to know basis.

Figure 12-4: System Logins and Uses

Login	UID	Use
root	0	Has no restrictions and overrides all other logins, protections, and permissions. It allows the user access to the entire system. The password for the root login should be very carefully protected.
sys	3	Owns many system files.
bin	2	Owns most of the commands.
adm	4	Owns certain administrative files.
uucp	5	Owns the object and spooled data files for uucp.
nuucp	10	Used by remote machines to log in to the system and start file transfers.
daemon	1	System daemon login; controls background processing.
lp	71	Owns the object and spooled data files for lp.

Assigning Special Administrative Passwords

After you have set up your system you should assign passwords to the special administrative and system logins. To do so, refer to the "System Setup" chapter.

Password Recovery

Limiting the number of people that know the root password is an important part of maintaining system security. Ideally, few people will know the password for this privileged login. However, when fewer people know the root password, the chances of losing or forgetting this password will increase. Make every effort to remember or discover the root password before performing this procedure.

Forgotten Root Password Recovery

If you cannot recover your root password, call your support hotline.

Forgotten Firmware Password Recovery

If you have forgotten your firmware password, refer to the hardware manual for your computer.

File Protection

Because the UNIX operating system is a multi-user system, you usually do not work alone in the file system. System users can follow pathnames to various directories and read and use files belonging to one another, as long as they have permission to do so.

If you own a file, you can decide who has the right to read it, write in it (make changes to it), or, if it is a program, to execute it. You can also restrict permissions for directories. When you grant execute permission for a directory, you allow the specified users to change directory to it and list its contents with the `ls(1)` command. Only the owner or a privileged user can define the following:

- which users have permission to access data
- which types of permission they have (that is, how they are allowed to use the data)

This section introduces types of files and discusses file protection.

File Types

When you display the contents of a directory with the `ls -l` command the first column of output describes the "mode" of the file. This information tells you not only what type of file it is, but who has permission to access it. This first field is 10 characters long. The first character defines the file type and can be one of the following types:

Figure 12-5: File Types

File Types	
Type	Symbol
Text, programs, etc.	-
Directories	d
Character special	c
Block special	b
FIFO (named pipe) special	p
Symbolic links	l

For more information on these types, see the "Storage Device Management" chapter or `ls(1)` in the *User's Reference Manual*.

File Access Permissions

In the first field of the `ls -l` output, the next nine characters are interpreted as three sets of three bits each. The first set refers to the owner's permissions; the next to permissions of members in the file's group; and the last to all others. Within each set, the three characters show permission to read, to write, and to execute the file as a program, respectively. For a directory, "execute" permission is interpreted to mean permission to search the directory for a specified file.

The permissions are as follows:

Figure 12-6: File Access Permissions

Permissions Explanation	Symbol
The file is readable.	`r`
The file is writable.	`w`
The file is executable.	`x`
This permission is *not* granted.	`-`
Mandatory locking will occur during access. (The set-group-ID bit is on and the group execution bit is off.)	`l`
The set-user-ID or set-group-ID bit is on, and the corresponding user or group execution bit is also on.	`s`
The set-user-ID bit is on and the user execution bit is off.	`S`
The sticky and the execution bits for `other` are on.	`t`
The sticky bit is turned on, and the execution bit for `other` is off.	`T`

Figure 12-7: Directory Access Permissions

Permissions Explanation	Symbol
The directory is readable.	`r`
The directory may be altered (files may be added or removed).	`w`
The directory may be searched. (This permission is required to `cd` to the directory.)	`x`
File removal from a writable directory is limited to the owner of the directory or file unless the file is writable.	`t`

For more information, refer to `ls(1)` and `chmod(1)` in the *User's Reference Manual*.

Setting a default `umask`

When a file is created its default permissions are set. These default settings may be changed by placing an appropriate `umask` command in the system profile (`/etc/profile`).

Figure 12-8: `umask(1)` Settings for Different Security Levels

Level of Security	umask	Disallows
Permissive	0002	w for others
Moderate	0027	w for group, rwx for others
Severe	0077	rwx for group and others

Set-UID and Set-GID

The set-user identification (set-UID) and set-group identification (set-GID) bits must be used carefully. These bits are set through the chmod(1) command and can be specified for any executable file. When any user runs an executable file that has either of these bits set, the system gives the user the permissions of the owner (or group) of the executable.

System security can be compromised if a user copies another program onto a file with -rwsrwxrwx permissions. For example, if the switch user (su) command has the write access permission allowed for others, anyone can copy the shell onto it and get a password-free version of su with no sulog entry being made. Experience has shown that people who have had root permissions once, tend to keep such a file around "just in case." The following paragraphs provide a few examples of command lines that can be used to identify the files with a set-UID. A vigilant system administrator will check the system for potential problems periodically and investigate any unusual occurrences.

For more information about the set-UID and set-GID bits, see chmod(1) and chmod(2).

Check Set-UIDs

The following command line lists all set-UID programs owned by root. The results are saved in a file in /tmp. All mounted paths are checked by this command starting at /. Any surprises in the output should be investigated. Search time is dependent on the number of entries in the directory to be searched.

This program can be run for sys, bin, and mail, as well.

```
# find / -user root -perm -4000 -exec ls -ldb {} \; > /tmp/ckprm
# cat /tmp/ckprm
-r-sr-xr-x   1 root     bin         38836 Aug 10 16:16 /usr/bin/at
-r-sr-xr-x   1 root     bin         19812 Aug 10 16:16 /usr/bin/crontab
---s--x--x   1 root     sys         46040 Aug 10 15:18 /usr/bin/ct
-r-sr-xr-x   1 root     sys         12092 Aug 11 01:29 /usr/lib/mv_dir
-r-sr-sr-x   1 root     bin         33208 Aug 10 15:55 /usr/lib/lpadmin
-r-sr-sr-x   1 root     bin         38696 Aug 10 15:55 /usr/lib/lpsched
---s--x---   1 root     rar         45376 Aug 18 15:11 /usr/rar/bin/sh
-r-sr-xr-x   1 root     bin         12524 Aug 11 01:27 /usr/bin/df
-rwsr-xr-x   1 root     sys         21780 Aug 11 01:27 /usr/bin/newgrp
-r-sr-sr-x   1 root     sys         23000 Aug 11 01:27 /usr/bin/passwd
-r-sr-xr-x   1 root     sys         23824 Aug 11 01:27 /usr/bin/su
#
```

In this example, an unauthorized user (`rar`) has made a personal copy of `/usr/bin/sh` and has made it set-UID to `root`. This means that `rar` can execute `/usr/rar/bin/sh` and become the privileged user.

If you want to save this output for future reference, move the file out of `/tmp`.

Check Set-UIDs by File System

The command line entry in the example below shows the use of the `ncheck` command to examine the `/usr` file system (`/dev/dsk/c1d0s2`, assuming a single-disk system with default partitioning) for files with a set-UID. The normal output of the `ncheck -s` command includes special files. The `-F` tells `ncheck` that it should expect an `s5` File System Type. If you are using some other file system type, refer to the "File System Administration" chapter. The output of the modified `ncheck` is used as an argument to the `ls` command. In this example, the complete pathnames for the files start with `/usr`. `/usr` is not part of the `ncheck` output but must be added (using sed(1)) for the `ls` to work. The use of the `ls` command is possible only if the file system is mounted.

```
# ls -l `ncheck -F s5 -s /dev/dsk/c1d0s2 | cut -f2 | sed 's:^:/usr:'`
-r-sr-xr-x   1 root     bin        72579 Mar  3 07:25 /usr/bin/at
-r-sr-xr-x   1 root     bin        33608 Mar  3 07:25 /usr/bin/atq
-r-sr-xr-x   1 root     bin        23040 Mar  3 07:25 /usr/bin/atrm
-r-sr-xr-x   1 root     bin        28424 Mar  3 07:25 /usr/bin/crontab
---s--x--x   1 root     uucp       74762 Mar  6 11:15 /usr/bin/ct
---s--x--x   1 uucp     uucp       83346 Mar  6 11:15 /usr/bin/cu
-r-sr-xr-x   1 root     bin        29370 Mar  3 10:44 /usr/bin/df
-r-xr-sr-x   1 bin      sys        11990 Mar 14 12:34 /usr/sbin/fusage
-r-xr-sr-x   1 bin      sys        36068 Mar  3 01:37 /usr/bin/ipcs
-r-sr-xr-x   1 root     bin        34514 Mar  3 10:46 /usr/bin/login
-r-xr-sr-x   2 bin      mail       88724 Mar  3 10:46 /usr/bin/mail
-r-xr-sr-x   1 bin      mail       85034 Mar  3 10:54 /usr/bin/mailx
-rwsr-xr-x   1 root     sys         8718 Mar  3 10:44 /usr/bin/newgrp
-r-sr-sr-x   1 root     sys        21154 Mar  3 10:44 /usr/bin/passwd
-r-sr-xr-x   1 root     bin        24202 Mar  3 10:46 /usr/bin/ps
-r-xr-sr-x   2 bin      mail       88724 Mar  3 10:46 /usr/bin/rmail
-rwsr-xr-x   1 root     sys        17526 Mar  3 10:44 /usr/bin/sacadm
-r-xr-sr-x   1 bin      sys        39508 Mar  3 02:50 /usr/sbin/sadp
-r-sr-xr-x   1 root     root       35128 Mar 14 13:07 /usr/bin/su
---s--x--x   1 uucp     uucp       78668 Mar  6 11:15 /usr/bin/uucp
---s--x--x   1 uucp     uucp       36628 Mar  6 11:15 /usr/bin/uuglist
---s--x--x   1 uucp     uucp       32254 Mar  6 11:16 /usr/bin/uuname
---s--x--x   1 uucp     uucp       77550 Mar  6 11:16 /usr/bin/uustat
---s--x--x   1 uucp     uucp       81424 Mar  6 11:16 /usr/bin/uux
-r-xr-sr-x   1 bin      tty        14438 Mar  3 10:47 /usr/bin/write
-r-sr-xr-x   1 root     bin        20106 Mar  3 11:13 /usr/lib/mail/mail_pipe
-rwxr-sr-x   1 bin      mail        3268 Mar  3 11:04 /usr/lib/mailx/rmmail
-r-sr-xr-x   1 root     sys        15864 Mar  3 10:52 /usr/lib/mv_dir
---s--x--x   1 root     bin        26801 Mar  3 02:46 /usr/lib/pt_chmod
-r-xr-sr-x   1 bin      sys        16682 Mar  3 02:52 /usr/lib/sa/sadc
---s--x--x   1 uucp     uucp      150868 Mar  6 11:17 /usr/lib/uucp/uucico
---s--x--x   1 uucp     uucp       51748 Mar  6 11:17 /usr/lib/uucp/uusched
---s--x--x   1 uucp     uucp       93294 Mar  6 11:18 /usr/lib/uucp/uuxqt
-r-sr-xr-x   1 root     sys        23824 Mar 11 01:27 /usr/rar/bin/su
-r-xr-sr-x   1 bin      tty        17488 Mar  3 10:43 /usr/sbin/wall
-r-xr-sr-x   1 bin      sys        11274 Mar  3 09:25 /usr/sbin/whodo

#
```

In this example, the `/usr/rar/bin/su` should be investigated.

Security Audit

After the system has been fully configured, the system administrator should perform a check for SETUID/SETGID files and devices on `root` and `/usr` using one of the above procedures. The output from this should be saved on some medium (for example, on a floppy disk) and printed in hard-copy. Periodically, rerun the procedure, compare its results with the previous output, and investigate changes such as additions, deletions, or changes in size or date.

This is an example for an AT&T 3B2 Computer.

```
# make sure /usr is mounted!
ls -l `ncheck -F s5 -s /dev/dsk/c1d0s0 | cut -f2 | grep -v dev` > sec.audit
ls -l `ncheck -F s5 -s /dev/dsk/c1d0s2 | cut -f2 | sed -e 's:^:/usr:'` >> sec.audit
```

Quick Reference to Security Procedures

The following commands allow `root` to do the tasks described in this chapter.

- Display password and aging information for a user.

  ```
  passwd -s login_name
  ```

- Display password and aging information for all users.

  ```
  passwd -s -a
  ```

- Change a user's password.

  `passwd` *login_name*

- Turn on aging, set *min* and *max*.

  `passwd -x` *max* `-n` *min login_name*

- Force a user to change the password at the next login session.

  `passwd -f` *login_name*

- Turn off password aging.

  `passwd -x -1` *login_name*

- Set the warning period before a user's password is to expire to one week.

  `passwd -w 7` *login_name*

- Lock a user's login.

  `passwd -l` *login_name*

- Set the expiration date for a user.

  `usermod -e` *mm/dd/yy login_name*

- Set the number of days a login is allowed to be inactive before it is locked.

  `usermod -f` *n login_name*

- Display inactive, warning, and expiration login information for a user.

  `logins -a -l` *login_name*

- Turn on login logging.

  (See the five-step procedure in the section "Login Logging.")

- List all set-UID programs owned by root.

  ```
  find / -user root -perm -4000 -exec ls -ldb {} \;
  ```

- List all files with a set-UID for the root file system.

  ```
  ncheck -s /dev/dsk/c1d0s0
  ```

- List all files with a set-UID for the `usr` file system.

  ```
  ncheck -s /dev/dsk/c1d0s2
  ```

13 Service Access

Introduction

This chapter is about managing access to system services, both networked and local. Beginning with UNIX System V Release 4, the Service Access Facility generalizes the procedures for service access so that login access on the local system and network access to local services are managed in essentially similar ways.

As part of this simplification, the port monitor `ttymon` has replaced `getty` and `uugetty` for local access to login service – although `ttymon` is not limited to login access.

Systems may access services using a variety of port monitors, including port monitors written expressly for a user's application. The section on the listener describes the administrative tasks required to provide access over any network that conforms to the Transport Interface (TLI) protocol. (TLI is documented in the *Programmer's Guide: Networking Interfaces*.)

The chapter is divided into six sections. The first, "Overview of the Service Access Facility," describes the Service Access Facility (SAF), its directory structure, its controlling program (the Service Access Controller or SAC), the configuration files that may be used to change the environment under which the SAC, port monitors, and services operate, and the SAF's two administrative files.

The second section, "Port Monitor Management," describes the SAC administrative command `sacadm` and how it is used to manage the information in the SAC administrative file. This includes:

- printing port monitor status information
- adding a port monitor to the system
- enabling or disabling a port monitor
- starting or stopping a port monitor, and
- removing a port monitor from the system

This section also describes the per-system and per-port monitor configuration scripts used to modify the SAC and port monitor environments and shows how to print, install, and modify each type of script.

A summary of the commands used to administer a port monitor is included at the end of the section.

The third section, "Service Management," follows the same outline. It describes the port monitor administrative command, pmadm. pmadm manages the information in the port monitors' administrative files. These files, one for each port monitor, include administrative and state information for each port and information about the service each port invokes. The pmadm command allows the system administrator to

- print information derived from the administrative file
- add and remove services
- enable and disable services
- print, install, and change per-service configuration scripts

A command summary is included at the end.

The fourth, fifth, and sixth sections describe two port monitors, ttymon and the listener. The section on ttymon describes what ttymon does and shows how to perform some of the administrative tasks described in the sections on port monitor management and service management—this time from the point of view of ttymon. These tasks include:

- listing the ttymon port monitors that have been configured
- listing the services that have been configured under a given ttymon port monitor
- enabling and disabling ttymon ports and services
- adding and removing a ttymon port monitor
- adding a service under a ttymon port monitor

The management of terminal line settings is treated with ttymon management, since the ttydefs file, which replaces gettydefs as the database file for system TTY terminal information, is used by ttymon and will be modified by the system administrator in the process of ttymon administration. Setting terminal modes and line speeds includes:

- printing terminal line setting information
- modifying terminal line settings

- setting up hunt sequences
- adding and removing terminal line settings for a terminal
- terminal options available using the `stty` command

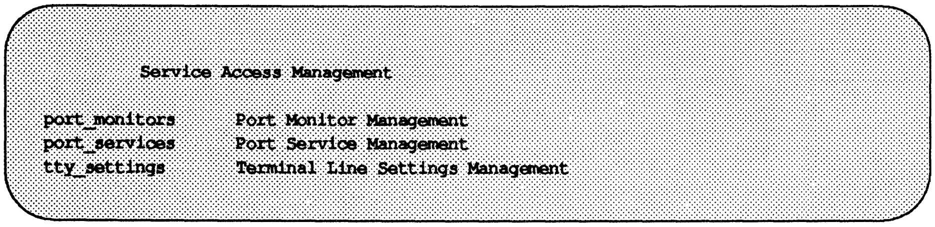
A summary of commands used in `ttymon` and terminal line settings administration is included at the end of the section.

The sixth and last section of the chapter describes the listener, what it does, and how to perform listener-related administrative tasks. These include:

- listing configured `listen` port monitors
- listing services available through a given `listen` port monitor
- enabling and disabling `listen` ports and services
- adding and removing `listen` port monitors
- adding a `listen` service

Tasks associated with port and service administration may be performed using either the menu system or shell commands entered on the command line. The screen shown below is the top-level menu for these functions. It can be invoked by typing `sysadm ports`:

Figure 13-1: Top-level menu for port monitor, service access, and terminal line settings management.



```
          Service Access Management

port_monitors       Port Monitor Management
port_services       Port Service Management
tty_settings        Terminal Line Settings Management
```

When you have selected one of the options, the submenus and instructions displayed on the screen should be self-explanatory and will lead you through the appropriate procedures.

The Port Monitor Management and Port Service Management menus provide a generalized and user-friendly interface for administering many types of ports. The Terminal Line Setting Management menu concerns only TTY ports and is helpful in defining the default `termio` settings.

The shell commands corresponding to the first two of these screen options are described in the sections of this chapter headed "Port Monitor Management" and "Service Management." The commands corresponding to the third are described under "The Port Monitor `ttymon`" in the section headed "Terminal Line Settings."

Overview of the Service Access Facility

As networking services have been added to UNIX System V, unanticipated methods of access to system services have evolved resulting in inconsistent interfaces and execution environments for service users. The Service Access Facility (SAF) provides a mechanism for uniform access to services. From the system administrator's point of view, the main components of this generalized access procedure are the commands for installing, configuring, and maintaining port monitors and services and the administrative or database files in which port monitor and service information is stored.

The manner in which a port monitor monitors and manages access ports is specific to the port monitor and not to any component of the Service Access Facility. Users may therefore extend their systems by developing and installing their own port monitors. It is one of the important features of the Service Access Facility that it can be extended in this way by users. Users who want to write their own port monitors should see the *Programmer's Guide: Networking Interfaces.* The present chapter limits its description of specific port monitors to those delivered with UNIX System V, `ttymon` and the listener.

From the point of view of the Service Access Facility, a service is a process that is started. There are no restrictions on the functions a service may provide.

The Service Access Facility consists of a controlling process, the Service Access Controller (SAC), and two administrative levels corresponding to two levels in the supporting directory structure. The top administrative level is concerned with port monitor administration, the lower level with service administration.

From an administrative point of view, the Service Access Facility consists of the following components, each of which is described in this document:

- The Service Access Controller
- A per-system configuration script
- The SAC administrative file
- The SAC administrative command `sacadm`
- Port monitors
- Optional per-port monitor configuration scripts
- An administrative file for each port monitor

- The administrative command pmadm
- Optional per-service configuration scripts

The Service Access Controller, the administrative files, and the per-system, per-port monitor, and per-service configuration files are described in this section. The administrative command sacadm is described under "Port Monitor Management" and the administrative command pmadm is described under "Service Management."

The Service Access Controller

The Service Access Controller is the overseer of the server machine. It is the Service Access Facility's controlling process. The SAC is started by `init(1M)` by means of an entry in `/sbin/inittab`. Its function is to maintain the port monitors on the system in the state specified by the system administrator. These states include: STARTING, ENABLED, DISABLED, STOPPING, NOTRUNNING, and FAILED. (A port monitor enters the FAILED state if the SAC cannot start it after a specified number of tries.)

The administrative command sacadm is used to tell the SAC to change the state of a port monitor. sacadm can also be used to add or remove a port monitor from SAC supervision and to list information about port monitors known to the SAC.

The SAC's administrative file contains a unique tag for each port monitor known to the SAC and the pathname of the command used to start each port monitor.

The SAC performs three main functions. Briefly:

- it customizes its own environment
- it starts the appropriate port monitors
- it polls its port monitors and initiates recovery procedures when necessary

During initialization, the SAC customizes its own environment by invoking the per-system configuration script. Next, it reads its administrative file to determine which port monitors are to be started. For each port monitor it starts, it interprets the port monitor's configuration script, if one exists. Finally the port monitors specified in the administrative file (for example, `ttymon`) are started.

Once the port monitors are running, the SAC polls them periodically for status information. The sac(1M) command-line option, -t, allows the system administrator to control polling frequency. When the port monitor gets a request for status from the SAC, it must respond with a message containing its current state (e.g., ENABLED). If the SAC does not receive a response, it assumes the port monitor is not running. If a port monitor that should be running has stopped, the SAC assumes it has failed and takes appropriate recovery action.

The SAC will restart a failed port monitor if a non-zero restart count was specified for the port monitor when it was created (see the sacadm command, described under "Port Monitor Management," below, and on the sacadm(1M) manual page in the *System Administrator's Reference Manual.*)

The SAC is the administrative point of control for all port monitors (and therefore for all ports on the system). The administrative commands sacadm(1M) and pmadm(1M) pass requests to the SAC, which in turn communicates with the port monitors. These requests include enabling a disabled port monitor so that it begins accepting service requests on its ports; starting port monitors that were previously killed; and listing the current state of all port monitors on the system.

The Per-System Configuration File

The per-system configuration file, /etc/saf/_sysconfig, is delivered empty. It may be used by the system administrator to customize the environment for all services on the system by writing a command script in the interpreted language described in the *Programmer's Guide: Networking Interfaces* and on the doconfig(3N) manual page in the *System Administrator's Reference Manual.* The per-system configuration script is interpreted by the Service Access Controller when the SAC is started. The SAC is started when the system enters multi-user mode.

Per-Port Monitor Configuration Scripts

Per-port monitor configuration scripts (`/etc/saf/`*pmtag*`/_config`) are optional. They allow the system administrator to customize the environment for any given port monitor and for the services that are available through the specific collection of access points for which that port monitor is responsible. Per-port monitor configuration scripts are written in the same language used for per-system configuration scripts.

The per-port monitor configuration script is interpreted when the port monitor is started. The port monitor is started by the Service Access Controller after the SAC has itself been started and after it has run its own configuration script, `/etc/saf/_sysconfig`.

The per-port monitor configuration script may override defaults provided by the per-system configuration script.

Per-Service Configuration Scripts

Per-service configuration files allow the system administrator to customize the environment for a specific service. For example, a service may require special privileges that are not available to the general user. Using the language described in the `doconfig`(3N) manual page, the system administrator can write a script that will grant or limit such special privileges to a particular service offered through a particular port monitor.

The per-service configuration may override defaults provided by higher-level configuration scripts. For example, the per-service configuration script may specify a set of STREAMS modules other than the default set.

The SAC Administrative File

The SAC's administrative file contains information about all the port monitors for which the SAC is responsible. This file exists on the delivered system. Initially, it is empty except for a single comment line which contains the version number of the Service Access Controller. The system administrator adds port monitors to the system by making entries in the SAC's administrative file. These entries are made using the administrative command `sacadm` with a –a option. `sacadm` is also used to remove entries from the SAC's administrative file.

If the software being installed adds the appropriate port monitor entry to the port monitor administrative file, the entry will not need to be added by the system administrator. For example, when the STARLAN package is installed, a `listen` port monitor named `starlan` is automatically created (that is, a line defining it is put in the SAC's administrative file by the package software).

Each entry in the SAC's administrative file contains the following information:

PMTAG
: A unique tag that identifies a particular port monitor. The system administrator is responsible for naming a port monitor. This tag is then used by the Service Access Controller (SAC) to identify the port monitor for all administrative purposes.

  PMTAG may consist of up to 14 alphanumeric characters.

PMTYPE
: The type of the port monitor. In addition to its unique tag, each port monitor has a type designator. The type designator identifies a group of port monitors that are different invocations of the same entity. `ttymon` and `listen` are examples of valid port monitor types. The type designator is used to facilitate the administration of groups of related port monitors. Without a type designator, the system administrator has no way of knowing which port monitor tags correspond to port monitors of the same type.

  PMTYPE may consist of up to 14 alphanumeric characters.

FLGS
: The flags that are currently defined are:

  d   When started, do not enable the port monitor.

  x   Do not start the port monitor.

  If no flag is specified, the default action is taken. By default a port monitor is started and enabled.

RCNT
: The number of times a port monitor may fail before being placed in a failed state. Once a port monitor enters the failed state, the SAC will not try to restart it. If a count is not specified when the entry is created, this field is set to `0`. A restart count of 0 indicates that the port monitor is not to be restarted when it fails.

COMMAND A string representing the command that will start the port monitor. The first component of the string, the command itself, must be a full pathname.

The figure shows the contents of a sample SAC administrative file as listed by the sacadm command. The # character at the end of each line is a comment delimiter.

Figure 13-2: Output of `sacadm -l`.

```
PMTAG     PMTYPE   FLGS  RCNT  STATUS     COMMAND
starlan   listen   -     0     ENABLED    /usr/lib/saf/listen -m slan starlan \
                                             # starlan listener
ttymon1   ttymon   d     0     DISABLED   /usr/lib/saf/ttymon # ttymon1
ttymon3   ttymon   -     0     ENABLED    /usr/lib/saf/ttymon # ports board
```

The Port Monitor Administrative File

Each port monitor has its own administrative file. The pmadm command is used to add, remove, or modify entries in this file. Each time a change is made, the corresponding port monitor is told to reread its administrative file.

Each entry in a port monitor's administrative file defines how the port monitor should treat a specific port and what service is to be invoked on that port. Some fields must be present for all types of port monitors. Each entry must include a service tag to identify the service uniquely and an identity to be assigned to the service when it is started (for example, root).

NOTE The combination of a service tag and a port monitor tag uniquely define an instance of a service. The same service tag may be used to identify a service under a different port monitor.

The record must also contain port monitor specific data, such as the prompt string, which are meaningful to `ttymon`. In general, each type of port monitor provides a command that takes the necessary port monitor-specific data as arguments and outputs these data in a form suitable for storage in the file. The `ttyadm`(1M) command does this for `ttymon` and `nlsadmin`(1M) does it for `listen`.

NOTE If the software being installed adds the appropriate service entries to the port monitor administrative file, the entries will not have to be added by the system administrator. For example, when Remote File Sharing (RFS) is installed, the package software installs the appropriate service under each `listen`-type port monitor.

Each entry in the port monitor administrative file must contain the following information.

SVCTAG	A unique tag that identifies a service. This tag is unique only for the port monitor through which the service is available. Other port monitors may offer the same or other services with the same tag. A service requires both a port monitor tag and a service tag to identify it uniquely. SVCTAG may consist of up to 14 alphanumeric characters.
FLGS	Flags with the following meanings may currently be included in this field: x Do not enable this port. By default the port is enabled. u Create a `utmp` entry for this service. By default no `utmp` entry is created for the service. Note that port monitors may ignore the u flag if creating a `utmp` entry for the service is not appropriate to the manner in which the service is to be invoked. Some services may not start properly unless `utmp` entries have been created for them (for example, `login`).
ID	The identity under which the service is to be started. The identity has the form of a login name as it appears in `/etc/passwd`.
PMSPECIFIC	Examples of port monitor-specific information are addresses, the name of a process to execute, or the name of a STREAMS pipe to pass a connection through.
COMMENT	A comment associated with the service entry.

NOTE

Each port monitor administrative file must contain one special comment of the form:

```
# VERSION=value
```

where *value* is an integer that represents the port monitor's version number. The version number defines the format of the port monitor administrative file. This comment line is created automatically when a port monitor is added to the system. It appears on a line by itself, before the service entries.

The figure shows lines from a sample `ttymon` administrative file. Note that everything in the PMSPECIFIC column is specific to a `ttymon` port monitor. The listing for a `listen` administrative file, for example, will contain a different set of entries in this column. Port-monitor specific information is formatted by the port monitor's administrative command, in this case `ttyadm`. The `ttyadm` command is included as part of the `pmadm` command when it is used with the `-a` option. See "Adding a Service," under "Service Management," below.

The figure shows the contents of a sample `ttymon` administrative file as listed by the `pmadm` command. The `#` character is a comment delimiter.

Figure 13-3: Output of `pmadm -l -p ttymon3`.

```
PMTAG    PMTYPE  SVCTAG  FLGS  ID    PMSPECIFIC
ttymon3  ttymon  31      ux    root  /dev/term/31 - - /usr/bin/login - 9600 - login: - #/dev/term/31
ttymon3  ttymon  32      ux    root  /dev/term/32 - - /usr/bin/login - 9600 - login: - #/dev/term/32
ttymon3  ttymon  33      ux    root  /dev/term/33 - - /usr/bin/login - 9600 - login: - #/dev/term/33
ttymon3  ttymon  34      ux    root  /dev/term/34 - - /usr/bin/login - 9600 - login: - #/dev/term/34
```

NOTE

To maintain the integrity of the system, it is strongly recommended that changes in the SAC and port monitor administrative files be made with the `sacadm` and `pmadm` commands, not by editing the files. The SAC does not recognize changes in some of the fields in these files unless they are made using the appropriate administrative command. Editing the file directly can lead to unexpected results.

Port Monitor Management

The Service Access Facility administrative model is hierarchical. The highest level is concerned with port monitor administration. The lower level is concerned with service administration and is discussed under "Service Management," below. At the level of port monitor administration, port monitors may be added, removed, started, stopped, enabled, or disabled. Other functions performed at this level include requesting port monitor status information, replacing a per-system configuration file, installing or replacing a per-port monitor configuration file, and requesting that a port monitor read its administrative file.

They are discussed at the end of this section. Configuration files are described under "Printing, Installing, and Replacing Configuration Scripts." Requesting that a port monitor read its administrative file is under "Reading the Administrative Files."

The SAC Administrative Command `sacadm`

`sacadm` is the administrative command for the upper level of the Service Access Facility hierarchy, that is, for port monitor administration (see the manual page `sacadm`(1M) in the *System Administrator's Reference Manual*). Under the Service Access Facility, port monitors are administered by using the `sacadm` command to make changes in the SAC's administrative file. `sacadm` performs the functions listed below. Each function is discussed in one of the following sections.

- print the requested port monitor information from the SAC administrative file
- add or remove a port monitor
- enable or disable a port monitor
- start or stop a port monitor
- install or replace a per-system configuration script
- install or replace a per-port monitor configuration script
- ask the SAC to reread its administrative file

Printing Port Monitor Status Information

```
sacadm -L [ -p pmtag | -t type ]
sacadm -l [ -p pmtag | -t type ]
```

pmtag is the tag associated with the port monitor that is being listed. *type* specifies the port monitor type, for example, `listen`. Unless the system administrator already knows the type of a port monitor, it may be necessary to use the most general form of the command (`sacadm -l`) to find out what the valid type and tag names are.

The `-l` and `-L` options request port monitor status information and may be invoked by any user on the system. The `-l` by itself lists status information for all port monitors on the system. The `-l` option with a `-p` option lists status information for port monitor *pmtag*. A `-l` with `-t` lists status information for all port monitors of type *type*. Any other combination of options with the `-l` option is invalid.

The `-L` option is identical to the `-l` option except that its output is printed in a condensed format.

Options that request information write the requested information to the standard output. A request for information using the `-l` option prints column headers and aligns the information under the appropriate headings. A request for information in the condensed format using the `-L` option prints the information in colon-separated fields. If the `-l` option is used, empty fields are indicated by a hyphen. If the `-L` option is used, empty fields are indicated by two successive colons.

The following sample output shows the differences between some of the options described above. The command

```
sacadm -l
```

lists status information for all port monitors:

Figure 13-4: Sample output of sacadm -l. This is the most general form of the list option.

```
PMTAG     PMTYPE   FLGS  RCNT  STATUS     COMMAND
starlan   listen   -     0     ENABLED    /usr/lib/saf/listen -m slan starlan \
                                             # starlan listener
ttymon1   ttymon   d     0     DISABLED   /usr/lib/saf/ttymon # ttymon1
ttymon3   ttymon   -     0     ENABLED    /usr/lib/saf/ttymon # ports board
```

Note that if ttymon1 is enabled (`sacadm -e -p ttymon1`), the entry in the STATUS field changes from DISABLED to ENABLED, but the entry in the FLGS field does *not* change. The `d` flag indicates that the port monitor goes immediately to DISABLED state *when it is started.* After it has been started, the system administrator can put it in ENABLED state. The flags field conveys information about the state in which a port monitor *starts,* not about its current state.

The command

```
sacadm -l -p starlan
```

lists status information only for port monitor `starlan`:

Figure 13-5: Sample output of sacadm -l when a port monitor is specified.

```
PMTAG     PMTYPE   FLGS  RCNT  STATUS     COMMAND
starlan   listen   d     3     DISABLED   /usr/lib/saf/listen -m slan starlan \
                                             # starlan listener
```

The same command using `-L` instead of `-l` will produce:

Figure 13-6: Sample sacadm output when the -L option is used. The -L option prints status information in a condensed format.

```
starlan:listen:d:3:DISABLED:/usr/lib/saf/listen -m slan starlan # starlan listener
```

The command

```
sacadm -l -t ttymon
```

lists status information for all port monitors whose type is ttymon:

Figure 13-7: Status information for port monitors of a single type.

```
PMTAG     PMTYPE   FLGS  RCNT  STATUS      COMMAND
ttymon1   ttymon   d     0      DISABLED    /usr/lib/saf/ttymon # ttymon1
ttymon3   ttymon   -     0      ENABLED     /usr/lib/saf/ttymon # ports board
```

Adding a Port Monitor

sacadm -a -p *pmtag* -t *type* -c "*cmd*" -v *ver* [-f dx] [-n *count*] \
    [-y "*comment*"] [-z *script*]

sacadm with a -a option is used by the system administrator or by a package that is being installed to create new instances of a port monitor. Because of the complexity of the options and arguments that follow the -a option, it may be advisable for the system administrator to use a command script or the menu system to add port monitors. To use the menu system, type sysadm ports and then choose the port_monitors option.

When sacadm creates a port monitor, it creates the supporting directory structure in /etc/saf and /var/saf for the new port monitor *pmtag* and the port monitor administrative file. It also adds an entry for the new port monitor to the SAC's administrative file.

The options following the -a option have the following meanings:

The -c option is followed by a command enclosed in double quotes. This is the command the SAC executes to start the port monitor.

The -v option is followed by the version number of the port monitor. The version number may be given to sacadm by the port monitor's special administrative command, as an argument to the -v option. For example:

```
    -v `ttyadm -V`
```

The port monitor-specific command is ttyadm for ttymon and nlsadmin for listen (see the manual pages ttyadm(1M) and nlsadmin(1M)). The version stamp of the port monitor is known by the command and is returned when the port monitor administrative command is invoked with the -V option. The version number is added to the new administrative file as a comment line of the form

```
    # VERSION=value
```

where *value* is an integer that represents the port monitor version number. The version number defines the file format. It provides a means of synchronizing software releases of port monitors with their properly formatted administrative files.

The -f option specifies one or both of the two flags d and x. The flags have the following meanings:

d	Do not enable the port monitor
x	Do not start the port monitor

If the -f option is not included in the command line no flags are set and the default conditions prevail. By default a port monitor is started and enabled.

The -n option sets the restart count to *count*. If a restart count is not specified when adding a port monitor, *count* is set to 0. A count of 0 indicates that the port monitor is not to be restarted if it fails.

The -y option includes "*comment*" in the SAC administrative file entry for the port monitor being added.

The -z option names a file whose contents are installed as the per-port monitor configuration script, _config.

The command line in the following figure adds a STARLAN port monitor of type `listen`.

Figure 13-8: Adding a `listen` port monitor.

```
sacadm -a -p starlan -t listen -c "/usr/lib/saf/listen -m slan starlan" \
         -v `nlsadmin -V`
```

Figure 13-9: Adding a `ttymon` port monitor.

```
sacadm -a -p ttymon1 -t ttymon -c "/usr/lib/saf/ttymon" -v 'ttymon -V'
```

Enabling, Disabling, Starting, and Stopping a Port Monitor

sacadm -e -p *pmtag*
sacadm -d -p *pmtag*
sacadm -s -p *pmtag*
sacadm -k -p *pmtag*

The syntax for these four command lines is the same, with -e, -d, -s, and -k standing for the mnemonically intelligible "enable," "disable," "start," and "kill."

sacadm with a -e option enables a port monitor. The SAC sends an enable message to the port monitor.

The -d option disables a port monitor. The SAC sends a disable message to the port monitor.

sacadm with a -s option tells the SAC to start a port monitor.

The -k option stops or "kills" a port monitor. The SAC sends the signal SIGTERM to the port monitor.

Removing a Port Monitor

```
        sacadm -r -p pmtag
```

The commands that enable/disable and start/stop port monitors form command pairs, each of which reverses the action of the other. The -r option can be thought of as part of such a pair with the -a option. sacadm with a -r removes port monitor *pmtag* from the system. The port monitor entry is removed from the SAC's administrative file and the SAC rereads the file. If the removed port monitor is not running, no further action is taken. If the removed port monitor is running, the Service Access Controller sends it SIGTERM to indicate that it should shut down. Note that the port monitor's directory structure remains intact but is no longer referenced by anything.

Printing, Installing, and Replacing Configuration Scripts

Per-system and per-port monitor configuration scripts are administered using sacadm; per-service configuration scripts are administered using pmadm and are described under "Service Management" below. Per-system and per-port monitor configuration scripts allow the system administrator to modify the system and port monitor environments. They are written in the interpreted language described in the manual page doconfig(3N) and in the *Programmer's Guide: Networking Interfaces*. Sample configuration scripts are shown below.

The per-system configuration script, `_sysconfig`, is interpreted when the SAC is starting. A port monitor's per-port monitor configuration script is interpreted by the SAC just before the SAC starts the port monitor.

Per-system and per-port monitor configuration scripts may be printed by any user on the system. Only the system administrator may install or replace them.

Per-System Configuration Scripts

`sacadm -G [ -z` *script* `]`

The per-system configuration script `/etc/saf/_sysconfig` customizes the environment for all services on the system. When it starts up, the Service Access Controller interprets the per-system configuration script, using the `doconfig` library routine. A default `_sysconfig` containing only a comment line is part of the delivered system.

The `-G` option is used to print or replace the per-system configuration script. The `-G` option by itself prints the per-system configuration script. The `-G` option in combination with a `-z` option replaces `/etc/saf/_sysconfig` with the contents of the file *script*. Other combinations of options with a `-G` option are invalid.

The `_sysconfig` file in the figure sets the time zone variable, `TZ`.

Figure 13-10: Sample per-system configuration script.

```
assign TZ=EST5EDT          # set TZ
runwait echo SAC is starting > /dev/console
```

NOTE

The `-z` option is also used with the `-a` option to specify the contents of the per-port monitor configuration file when a port monitor is created.

Per-Port Monitor Configuration Scripts

`sacadm -g -p` *pmtag* `[ -z` *script* `]`

The per-port monitor configuration script `/etc/saf/`*pmtag*`/_config` customizes the environment for services that are available through the specific collection of access points for which port monitor *pmtag* is responsible. When the SAC starts a port monitor, the per-port monitor configuration script is interpreted, if it exists, using the `doconfig`(3N) library routine.

The `-g` option is used to print, install, or replace a per-port monitor configuration script. A `-g` option requires a `-p` option. The `-g` option with only a `-p` option prints the per-port monitor configuration script for port monitor *pmtag*. The `-g` option with a `-p` option and a `-z` option installs the file *script* as the per-port monitor configuration script for port monitor *pmtag*, or, if `/etc/saf/`*pmtag*`/_config` exists, it replaces `_config` with the contents of *script*. Other combinations of options with `-g` are invalid.

In the hypothetical `_config` file in the figure, the command `/usr/bin/daemon` is assumed to start a daemon process that builds and holds together a STREAMS multiplexor. By installing this configuration script, the command can be executed just before starting the port monitor that requires it.

Figure 13-11: Sample per-port monitor configuration script.

```
run /usr/bin/daemon
# build a STREAMS multiplexor.
runwait echo $PMTAG is starting > /dev/console
```

Reading the Administrative Files

```
sacadm -x [ -p pmtag ]
```

When changes are made to the SAC's administrative file, the SAC needs to be notified of the change. When changes are made to a port monitor's administrative files, the port monitor needs to notified. When `sacadm` and `pmadm` are used to make changes, this notification takes place automatically. If the files are edited by the system administrator, the SAC and the port monitors are not notified. In this case, `sacadm` must be called with the `-x` option to notify the SAC or port monitor of the changes.

`sacadm` with the `-x` option tells the SAC to update its internal copy of the information in the SAC administrative file. `sacadm` with the `-x` and `-p` options causes the SAC to send a READ message to the designated port monitor.

System administrators are advised against editing these files directly.

Quick Reference to Port Monitor Management

COMMAND SYNTAX	DESCRIPTION
sacadm -a -p *pmtag* -t *type* -c "*cmd*" -v *ver* [-f dx] [-n *count*] \ [-y "*comment*"] [-z *script*]	Add a port monitor entry to the SAC's administrative file.
sacadm -l [-p *pmtag* | -t *type*]	Print port monitor status information.
sacadm -L [-p *pmtag* | -t *type*]	Print port monitor status information in condensed format.
sacadm -G [-z *script*]	Print or replace per-system configuration script /etc/saf/_sysconfig.
sacadm -g -p *pmtag* [-z *script*]	Print or replace per-port monitor configuration script /etc/saf/*pmtag*/_config.
sacadm -e -p *pmtag*	Enable port monitor *pmtag*.
sacadm -d -p *pmtag*	Disable port monitor *pmtag*.
sacadm -s -p *pmtag*	Start port monitor *pmtag*.
sacadm -k -p *pmtag*	Stop port monitor *pmtag*.
sacadm -r -p *pmtag*	Remove the entry for port monitor *pmtag* from the SAC administrative file.

Service Management

The top level of the Service Access Facility is concerned with port monitor administration and is discussed in the section headed "Port Monitor Management" above. The lower level is concerned with service administration and is discussed in this section.

At this level there are two distinct administrative functions. The first is the administration of the port itself. The information needed to administer a port will be found on the manual page for `ttymon`'s port monitor-specific command, `ttyadm`(1M). The information needed to administer a network address monitored by a `listen` port monitor will be found on the manual page for `listen`'s port monitor-specific command, `nlsadmin`(1M).

The second is the administration of the service associated with a port. By definition, there is one and only one service associated with a port. All ports on the system are peers and their services are administered through the same command interface, the Service Access Facility's administrative command `pmadm`(1M). At the level of service administration, services may be added, removed, enabled, and disabled. Other functions performed at this level include installing or replacing a per-service configuration script and requesting service status information.

The Port Monitor Administrative Command `pmadm`

`pmadm` is the administrative command for the lower level of the Service Access Facility hierarchy, that is, for service administration. A port may have only one service associated with it although the same service may be available through more than one port. `pmadm` performs the following functions:

- print service status information from the port monitor's administrative file
- add or remove a service
- enable or disable a service
- install or replace a per-service configuration script

Note that in order to identify an instance of a service uniquely, the `pmadm` command must identify both the service (`-s`) and the port monitor or port monitors through which the service is available (`-p` or `-t`).

Printing Service Status Information

```
pmadm -l [ -t type | -p pmtag ] [ -s svctag ]
pmadm -L [ -t type | -p pmtag ] [ -s svctag ]
```

The `-l` and `-L` options request service status information. They may be invoked by any user on the system. Used either alone or with the options described below they provide a filter for extracting information in several different groupings.

`-l`	By itself, the `-l` option lists status information for all services on the system.
`-l -p` *pmtag*	Lists status information for all services available through port monitor *pmtag*.
`-l -s` *svctag*	Lists status information for all services with the tag *svctag* available through any port monitor on the system.
`-l -p` *pmtag* `-s` *svctag*	Lists status information for service *svctag* available through port monitor *pmtag*.
`-l -t` *type*	Lists status information for all services available through port monitors of type *type*.
`-l -t` *type* `-s` *svctag*	Lists status information for all services with the tag *svctag* offered through a port monitor of type *type*.

Other combinations of options with `-l` are invalid.

The `-L` option is identical to the `-l` option except that output is printed in a condensed format.

Options that request information write the requested information to the standard output. A request for information using the `-l` option prints column headers and aligns the information under the appropriate headings. A request for information in the condensed format using the `-L` option prints the information in colon-separated fields. If the `-l` option is used, empty fields are indicated by a hyphen. If the `-L` option is used, empty fields are indicated by two successive colons.

Figure 13-12: Output of pmadm -l.

```
PMTAG    PMTYPE  SVCTAG FLGS ID     PMSPECIFIC
ttymon3  ttymon  31     ux   root   /dev/term/31 - - /usr/bin/login - 9600 - login: - #/dev/term/31
ttymon3  ttymon  32     ux   root   /dev/term/32 - - /usr/bin/login - 9600 - login: - #/dev/term/32
ttymon3  ttymon  33     ux   root   /dev/term/33 - - /usr/bin/login - 9600 - login: - #/dev/term/33
ttymon3  ttymon  34     ux   root   /dev/term/34 - - /usr/bin/login - 9600 - login: - #/dev/term/34
ttymon1  ttymon  11     ux   root   /dev/term/11 - - /usr/bin/login - 9600 - login: - #/dev/term/11
ttymon1  ttymon  12     ux   root   /dev/term/12 - - /usr/bin/login - 9600 - login: - #/dev/term/12
ttymon1  ttymon  13     ux   root   /dev/term/13 - - /usr/bin/login - 9600 - login: - #/dev/term/13
ttymon1  ttymon  14     ux   root   /dev/term/14 - - /usr/bin/login - 9600 - login: - #/dev/term/14
starlan  listen  101    -    listen  - c - /usr/lib/uucp/uucico -r0 -unuucp -iTLI \
                                          #UUCP access direct to server
starlan  listen  102    -    listen  - c - /usr/slan/lib/ttysrv -d -n ntty,tirdwr,ld0 \
                                          #UUCP access to server via login
starlan  listen  1000   -    listen  - c - /usr/slan/lib/tstserver #TP TEST SERVER
starlan  listen  105    -    listen  - c - /usr/net/servers/rfs/rfsetup #RFS SERVER
starlan  listen  0      -    root    - c - sfbul.serve c - /usr/lib/saf/nlps_server #NLPS server
starlan  listen  1      -    listen  - c - sfbul c - /usr/slan/lib/ttysrv -m ntty,tirdwr,ld0 \
                                          #TTY SERVER
```

Adding a Service

pmadm -a [-p *pmtag* | -t *type*] -s *svctag* -i *id* -m "*pmspecific*" \
-v *ver* [-f xu] [-y "*comment*"] [-z *script*]

pmadm with a -a option adds a service by making an entry for the new service in the port monitor's administrative file. It is important to be aware that a service implies a port and that there is a one-to-one mapping between ports and instances of services. Because of the complexity of the options and arguments that follow the -a option, it may be advisable to use a command script or the menu system to add services. If you use the menu system, type `sysadm ports` and then choose the `port_services` option.

The following paragraphs describe the components of the command line for adding a service.

-p specifies the tag associated with the port monitor through which a service (specified as -s *svctag*) is available. *pmtag* and *svctag* are names chosen by the system administrator.

The -t option specifies the port monitor type. Port monitors are specified either by a -t or by a -p but not both. If -p is used, a service is added to a single port monitor, port monitor *pmtag*. If -t is used, instances of a service are added to all port monitors of type *type*.

The -s option specifies the service tag.

The -i option specifies the identity that is to be assigned to service *svctag* when it is started. ID must be an entry in /etc/passwd.

The -m option allows port monitor-specific options to be included on the -a command line. This information should be generated by using a port monitor-specific command, with whichever of its options are appropriate.

The -v option specifies the version number of the port monitor administrative file. For a port monitor of type listen, for example, the version number may be given as

```
        -v `nlsadmin -V`
```

The port monitor-specific command for ttymon is ttyadm(1M). The port monitor-specific command for listen is nlsadmin(1M). The version stamp of the port monitor is known by the port monitor-specific command and is returned when the command is invoked with a -V option.

The -f option specifies one or both of two flags which are then included in the flags field of the port monitor administrative file entry for the new service. The flags have these meanings:

x	Do not enable the service.
u	Create a utmp entry for the service.

If the -f option is not included on the -a command line, no flags are set and the default conditions prevail. By default, a new service is enabled and no utmp entry is created for it.

-y precedes a comment enclosed in double quotes. *comment* is included in the comment field for the service entry in the port monitor administrative file.

`-z` installs *script* as a configuration file.

The example adds a service with service tag `105` to all port monitors of type `listen`.

```
pmadm -a -s 105 -i root -t listen -v 'nlsadmin -V' \
        -m 'nlsadmin -a 105 -c /usr/net/servers/rfsetup'
```

Enabling or Disabling a Service

`pmadm -e -p` *pmtag* `-s` *svctag*
`pmadm -d -p` *pmtag* `-s` *svctag*

`pmadm` with the `-e` option enables a service. `x` is removed from the flags field in the entry for service *svctag* in the port monitor administrative file.

The `-d` option disables a service. `x` is added to the flags field in the entry for service *svctag* in the port monitor administrative file.

Removing a Service

`pmadm -r -p` *pmtag* `-s` *svctag*

`pmadm` with a `-r` removes service *svctag*. The entry for the service is removed from the port monitor administrative file.

Printing, Installing, and Replacing Per-Service Configuration Scripts

`pmadm -g -p` *pmtag* `-s` *svctag* [`-z` *script*]
`pmadm -g -s` *svctag* `-t` *type* `-z` *script*

Per-service configuration scripts are command scripts written in the interpreted language described in the `doconfig`(3N) manual page and in the *Programmer's Guide: Networking Interfaces*. They allow the system administrator to modify the environment in which a service executes. For example, the values of environment variables may be changed, STREAMS modules may be specified, or commands may be run.

Per-service configuration scripts are interpreted by the port monitor before the service is invoked.

NOTE The SAC interprets both its own configuration file, `_sysconfig`, and the port monitor configuration files. Only the per-service configuration files are interpreted by the port monitors.

Per-service configuration scripts may be printed by any user on the system. Only the system administrator may install or replace them.

The `-g` option is used to print, install, or replace a per-service configuration script. The `-g` option with a `-p` option and a `-s` option prints the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-p` option, a `-s` option, and a `-z` option installs the per-service configuration script contained in the file *script* as the per-service configuration script for service *svctag* available through port monitor *pmtag*. The `-g` option with a `-s` option, a `-t` option, and a `-z` option installs the file *script* as the per-service configuration script for service *svctag* available through any port monitor of type *type*. Other combinations of options with `-g` are invalid.

The following per-service configuration script does two things: It specifies the maximum file size for files created by a process by setting the process's `ulimit` to 4096. It also specifies the protection mask to be applied to files created by the process by setting `umask` to 077.

Figure 13-13: Sample per-service configuration script.

```
runwait ulimit 4096
runwait umask 077
```

Quick Reference to Service Administration

COMMAND SYNTAX	DESCRIPTION

`pmadm -a [ -p` *pmtag* `| -t` *type* `] -s` *svctag* `-i` *id* `-m "`*pmspecific*`" \`
`-v` *ver* `[ -f xu ] [ -y "`*comment*`" ] [ -z` *script* `]`
	Add a service entry to the port monitor administrative file.

`pmadm -l [ -t` *type* `| -p` *pmtag* `] [ -s` *svctag* `]`	Print service status information.
`pmadm -L [ -t` *type* `| -p` *pmtag* `] [ -s` *svctag* `]`	Print service status information in condensed format.
`pmadm -g -p` *pmtag* `-s` *svctag* `[ -z` *script* `]`	Print, install, or replace per-service configuration script for service *svctag* associated with port monitor *pmtag*.
`pmadm -g -s` *svctag* `-t` *type* `-z` *script*	Install or replace per-service configuration scripts for all services *svctag* associated with port monitors of type *type*.
`pmadm -e -p` *pmtag* `-s` *svctag*	Enable service *svctag* associated with port monitor *pmtag*.
`pmadm -d -p` *pmtag* `-s` *svctag*	Disable service *svctag* associated with port monitor *pmtag*.
`pmadm -r -p` *pmtag* `-s` *svctag*	Remove the entry for service *svctag* from the port monitor administrative file.

The Port Monitor `ttymon`

`ttymon` is a port monitor invoked by the Service Access Controller (SAC). The Service Access Controller is the Service Access Facility's controlling process. It is started by `init` when the system enters multi-user mode. One of the SAC's functions after it is started is to start all port monitors the system administrator has configured.

Beginning with UNIX System V Release 4, `ttymon` performs the functions that `getty` and `uugetty` performed in previous releases. Like `getty` and `uugetty`, `ttymon` sets terminal modes and line speeds for the port the user is connected to, allowing communication with the service associated with that port.

`ttymon` differs from `getty` and `uugetty` in several important ways:

- `ttymon` provides any service the system administrator configures. `getty` and `uugetty` provided only `login` service.
- Each invocation of `ttymon` can support multiple ports. `getty` and `uugetty` supported only one port per invocation.
- `ttymon` is a persistent process that continues to run after the service process is initiated. The `getty` and `uugetty` processes were replaced by the process of the service invoked.
- `ttymon` can take advantage of all STREAMS I/O capabilities.
- Line disciplines are configurable on a per-port basis.
- `ttymon` provides an optional autobaud facility that automatically determines the line speed of the hardware connected to any port monitored by a `ttymon` port monitor.

What `ttymon` Does

`ttymon` has three main functions:

- It initializes and monitors TTY ports.
- It sets terminal modes and line speeds for each port it monitors.

- It invokes the service associated with a given port whenever it receives a connection request on that port.

Each instance of ttymon has its own administrative file that specifies the ports to monitor and the services associated with each port. The file contains a *ttylabel* field that refers to a speed and TTY definition in the /etc/ttydefs file. See ttyadm(1M) for a description of the information specific to ttymon that is contained in a ttymon administrative file.

When a ttymon port monitor is started, it initializes all ports specified in its administrative file, pushes the specified STREAMS modules onto the ports, sets speed and initial termio(7) settings, and writes the prompt to the port. It then waits for user input.

A connection request is successful when at least one non-break character followed by a newline character is received from the port. If the service to be invoked is login, the newline character will be preceded by the users login name. A newline character will not be recognized unless the line speed of the port and the line speed of the device connected to the port are the same.

If an unreadable prompt is printed on the terminal, the user sends a BREAK to indicate that the port and device line speeds are not compatible. Each break indication will cause ttymon to hunt to the next *ttylabel* in /etc/ttydefs, adjusting its termio(7) values and reissuing the prompt.

On successful completion of the connection request, ttymon interprets the per-service configuration script, if one exists. It then invokes the service associated with the port. This service can be any service configured by the system administrator. A typical example is login.

ttymon has no interaction with its TTY ports while they are connected to a service. On completion of a service on a port, ttymon returns the port to its initial state.

The Autobaud Option

Autobaud allows the system to set the line speed of a given TTY port to the line speed of the device connected to the port without the user's intervention. Each time a service to be monitored by a ttymon port monitor is added, a *ttylabel* must be supplied (see "Adding a Service," below). If this *ttylabel* points to an entry in the /etc/ttydefs file that has an "A" in the autobaud field, ttymon will try to determine the proper line speed before printing the prompt.

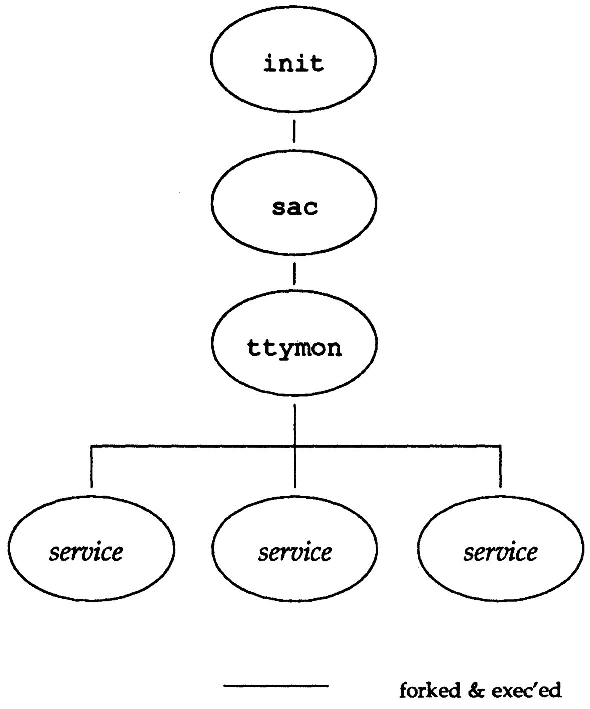
After receiving a carrier-indication on one of its TTY ports, but before printing a prompt, ttymon does the following:

- ttymon reads the next character received from the port. Provided the character read is a newline character and that it is transmitted at a line speed autobaud can support, ttymon will reliably determine this line speed and change the port's line speed to that speed.
- If a baud rate cannot be determined from the character that is read (for example, if the user entered a character other than a newline), or if a break is received rather than a character, ttymon considers this to be an autobaud failure and the character is discarded. If after five opportunities, a newline is not recognized, the search proceeds to the next ttydefs entry in the hunt sequence. If an autobaud flag is encounted again, the prompt will not be written and the procedure just described is repeated. If no autobaud flag is set, the search again proceeds to the next ttydefs entry in the hunt sequence.

ttymon and the Service Access Facility

The Service Access Facility (SAF) provides a generic interface to which all port monitors must conform. ttymon is a port monitor under the Service Access Facility's controller, the Service Access Controller. (See "Overview of the Service Access Facility," "Port Monitor Management," and "Service Management" for a description of the Service Access Facility, the administrative files it maintains, and the commands used for port monitor and service administration.) The figure below shows how a service, which may be a login service, is invoked using ttymon.

Figure 13-14: TTY service invocation.



There can be multiple invocations of `ttymon` port monitors, each identified by a unique *pmtag*. Each of these port monitors can monitor multiple ports for incoming connection requests.

A port has one and only one service associated with it. Each port, and its associated service, is identified by a service tag, *svctag*. Service tags for any given port monitor are unique.

When the Service Access Controller starts a port monitor, the port monitor reads its administrative file, which contains information about which ports to monitor and what service (i.e., process) is associated with each port.

The Default ttymon Configuration

Some ttymon port monitors may be set up automatically when the system goes to multi-user mode. To find out if your system has been automatically configured, enter the command

```
        sacadm -l
```

after the system is in multi-user mode. To see a listing of all services available under the configured ttymon port monitors, enter the command

```
        pmadm -l -t ttymon
```

The line discipline module, ldterm, may not be specified for automatically configured services. Instead, it may be defined in an autopush administrative file and pushed by the autopush facility (see the autopush(1M) manual page). Autopush pushes previously specified modules onto the appropriate STREAM each time a device is opened.

Services are not defined for the console and contty ports under any ttymon port monitor. Instead, there is an entry for each in the /sbin/inittab file. These entries contain calls to ttymon in "express" mode. (See "ttymon Express," below.)

The ttyadm Command

The Service Access Facility requires each type of port monitor to provide an administrative command. This command must format information derived from command-line options so that it is suitable for inclusion in the administrative files for that port monitor type. The command may also perform other port monitor-specific functions.

ttyadm is ttymon's administrative command. The ttyadm command formats information based on the options with which it is invoked and writes this information to the standard output.

ttyadm is one of the arguments pmadm uses with the -a option to format information in a way suitable for inclusion in a ttymon administrative file. ttyadm presents this information (as standard output) to pmadm, which places it in the file. This use of ttyadm is described below under "Adding a ttymon Port Monitor." *pmspecific* information in a port monitor administrative file will be different for different port monitor types.

ttyadm is also included on the sacadm command line when a port monitor is added to the system. It is used to supply the ttymon version number for inclusion in a port monitor's administrative file.

NOTE

The port monitor administrative file is updated by the Service Access Controller's administrative commands, sacadm and pmadm. ttyadm merely provides a means of presenting formatted port monitor-specific (i.e., ttymon-specific) data to these commands.

The sacadm command line uses ttyadm only with the -V option. ttyadm -V tells the SAC the version number of the ttymon command being used.

Managing TTY Ports

Finding Out What ttymon Port Monitors Are Configured

```
sacadm -l [ -p pmtag | -t type ]
```

The sacadm command with only a -l option lists all port monitors currently defined for the system. The following is an example of its output:

```
PMTAG    PMTYPE  FLGS RCNT   STATUS           COMMAND
starlan  listen  dx   5     NOTRUNNING  /usr/lib/saf/listen -m slan starlan  #
ttymon1  ttymon  -    0     ENABLED     /usr/lib/saf/ttymon                  #
ttymon2  ttymon  -    0     ENABLED     /usr/lib/saf/ttymon                  #
```

sacadm can also be used to list a single port monitor (-p) or to list only port monitors of a single type (-t), for example, all port monitors of type ttymon. For a complete description of these options, see "Printing Port Monitor Status Information" in the "Port Monitor Management" section above, or see the sacadm(1M) manual page.

Finding Out What Services Are Configured for a ttymon Port Monitor

pmadm -l [-p *pmtag* | -t *type*] [-s *svctag*]

pmadm with only a -l will list all services for all port monitors on the system. If a port monitor is specified (-p), all services for that port monitor will be listed. The following is a sample listing for the command

```
pmadm -l -p ttymon2
```

```
PMTAG    PMTYPE  SVCTAG FLGS ID   PMSPECIFIC
ttymon2  ttymon  21     ux   root /dev/term/21 — /usr/bin/login - 9600 - login: - #
ttymon2  ttymon  22     ux   root /dev/term/22 — /usr/bin/login - 9600 - login: - #
ttymon2  ttymon  23     ux   root /dev/term/23 — /usr/bin/login - 9600 - login: - #
ttymon2  ttymon  24     ux   root /dev/term/24 — /usr/bin/login - 9600 - login: - #
```

In the above table, the *pmspecific* fields include the device (for example, /dev/term/21), the service to be invoked (/usr/bin/login), and the prompt (login:). See the ttyadm(1M) manual page for a description of the *pmspecific* fields.

Finding Out Which TTY Ports Are Accessible

To find out which ports are accessible to users, first identify all enabled ttymon port monitors:

```
sacadm -l -t ttymon
```

```
#sacadm -l -t ttymon
PMTAG    PMTYPE  FLGS    RCNT    STATUS      COMMAND
ttymon1  ttymon  -       0       ENABLED     /usr/lib/saf/ttymon        #
ttymon3  ttymon  d       0       DISABLED    /usr/lib/saf/ttymon        #
```

In the listing, port monitor ttymon1 is enabled. This means that it is accepting service requests for any of its services that are enabled.

To identify which services are enabled, use

```
pmadm -l -p ttymonl
```

This will list all configured TTY services for port monitor `ttymonl`.

```
#pmadm -l -p ttymonl
PMTAG      PMTYPE  SVCTAG  FLGS  ID        <PMSPECIFIC>
ttymonl    ttymon  11      u     root      /dev/term/11 - - /usr/bin/login - 9600 - login: - #
ttymonl    ttymon  12      ux    root      /dev/term/12 - - /usr/bin/login - 9600 - login: - #
ttymonl    ttymon  13      u     root      /dev/term/13 - - /usr/bin/login - 9600 - login: - #
ttymonl    ttymon  14      ux    root      /dev/term/14 - - /usr/bin/login - 9600 - login: - #
```

In the listing, enabled services are those that do *not* have an `x` in the FLGS column. The ports corresponding to these services (`/dev/term/11` and `/dev/term/13`) are accessible to users.

NOTE On UNIX System V Release 4 `who -l` lists all running port monitors, not the accessible TTY ports. Follow the procedure described above to find out which TTY ports are accessible.

Adding a ttymon Port Monitor

sacadm -a -p *pmtag* -t *type* -c *cmd* -v \`*pmspecific* -V\` \\
  -n *count* [-f *dx*] [-z *script*] [-y *comment*]

The following command line will add a `ttymon`-type port monitor named `ttymonl`:

```
sacadm -a -p ttymonl -t ttymon -c /usr/lib/saf/ttymon \
       -v `ttyadm -V`
```

The command adds a line to the SAC's administrative file. The options that may be used with `sacadm -a` are described under "Port Monitor Management," above, and in the `sacadm`(1M) and `ttyadm`(1M) manual pages.

NOTE If a port monitor already exists with the same name as the port monitor that is being added, the system administrator must remove the old port monitor before adding the new one.

Removing a ttymon Port Monitor

```
        sacadm -r -p pmtag
```

The following command line removes the port monitor added in the previous example:

```
        sacadm -r -p ttymon1
```

The SAC removes the line for port monitor ttymon1 from its administrative file. The port monitor directory will remain in /etc/saf but will be removed and recreated when a new port monitor with the same name is added. To make changes to a port monitor entry, always remove the entry and add a new entry using the sacadm command. Do not edit the SAC administrative file.

Adding a Service

```
        pmadm -a -p pmtag -s svctag -i id [ -f ux ] -v `ttyadm -V` \
                -m "`ttyadm [ -b ] [ -r count] [ -c ] [ -h ] \
                    [-i msg] [-m modules] [-p prompt] [-t timeout] \
                     -d device -l ttylabel -s service`"
```

The following command line adds a login service to be monitored by the ttymon port monitor ttymon2:

```
        pmadm -a -p ttymon2 -s 21 -i root -fu -v `ttyadm -V` \
                -m "`ttyadm -d /dev/term/21 -l 9600 \
                -s /usr/bin/login -m ldterm -p \" tty21:\"`"
```

The options that may be used with pmadm -a are described under "Service Management," above, and on the pmadm(1M) and ttyadm(1M) manual pages.

The ttyadm -m option may be used for pushing STREAMS modules, for example the line discipline module, ldterm. If autopush has pushed modules on the stream, ttymon pops them before pushing its own.

By using the `ttyadm -i` option, we could also have specified a message to be printed whenever someone tries to log in on a disabled port.

The following command defines a service that permits both incoming and outgoing calls. The service is put under port monitor `ttymon2`. The `-b` option defines the port as bi-directional.

```
pmadm -a -p ttymon2 -s 21 -i root -fu -v `ttyadm -V` \
      -m "`ttyadm -b -h -r0 -t 60 -d /dev/term/21 \
      -l 9600H -s /usr/bin/login -m ldterm -p \" tty21:\"`"
```

The `ttyadm -r` option with count=0 is assumed when the `ttyadm -b` bi-directional option is used; the `-r0` could therefore have been omitted.

NOTE The Basic Networking Utilities package must be installed for bi-directional services.

Removing a Service

pmadm -r -p *pmtag* -s *svctag*

The following example deletes the service that was added in the previous example.

```
pmadm -r -p ttymon2 -s 21
```

Enabling a Service

pmadm -e -p *pmtag* -s *svctag*

To enable a service on a specific port, first find out which port monitor is monitoring the port. Enter

```
pmadm -l -t ttymon
```

This lists all services defined for ttymon-type ports.

Now look in the PMSPECIFIC column for the device file that corresponds to the port you are interested in, for example, `/dev/term/23`. If the port monitor is `ttymon2` and the service tag is `23`, the command

```
        pmadm -e -p ttymon2 -s 23
```

will enable the service on port /dev/term/23.

To verify that the port has been enabled, enter

```
        pmadm -l -p ttymon2 -s 23
```

The x will have been removed from the FLGS column in the entry for this service.

Disabling a Service

```
        pmadm -d -p pmtag -s svctag
```

When a service is disabled, all subsequent connection requests for the service will be denied. Using the same example,

```
        pmadm -d -p ttymon2 -s 23
```

will restore the x to the FLGS field in the entry for service 23.

Disabling All Services Monitored by a ttymon Port Monitor

```
        sacadm -d -p pmtag
```

To disable all services defined for the port monitor ttymon2, enter

```
        sacadm -d -p ttymon2
```

Any future connection requests for services managed by this port monitor will be denied until the port monitor is enabled.

The command

```
        sacadm -e -p ttymon2
```

will re-enable port monitor ttymon2.

ttymon "Express"

Services are not defined for the console and contty ports under any ttymon port monitor. Instead, there is an entry for each in the /sbin/inittab file. These entries contain calls to ttymon in "express" mode. ttymon express is a special mode of ttymon that permits ttymon to be invoked directly by a command that requires login service. ttymon in express mode is not managed by the Service Access Controller nor is an administrative file associated with any invocation of ttymon in this mode.

ttymon express is described in greater detail on the ttymon(1M) manual page.

Configuration Files

As a port monitor under the Service Access Facility, ttymon can customize the environment of each service it starts. It does this by interpreting a per-service configuration script, if one exists, immediately before starting the service. Per-service configuration scripts are optional. Configuration scripts are installed by the system administrator, using the pmadm command with -g and -z options (see the pmadm(1M) manual page).

It is also possible to customize the environment of a ttymon port monitor. A per-port monitor configuration script is defined using the sacadm command with -g and -z options (see the sacadm(1M) manual page). The environment modifications made by a port-monitor configuration script are inherited by the port monitor and all the services it invokes. The environment of any particular service can then be customized further by using a per-service configuration script.

The doconfig(3N) manual page describes the language in which configuration scripts are written.

Configuration scripts are not normally needed for basic operations.

The who Command

The who command examines the /var/adm/utmp file. It is used to find out who is on the system. The command

```
who -lH
```

lists all entries in the utmp file including all RUNNING port monitors. When ttymon in express mode is monitoring a line, the name field is LOGIN as it is in the entry for contty in the following example.

Figure 13-15: who -lH **output.**

```
#who -lH
NAME      LINE      TIME          IDLE  PID   COMMENTS
LOGIN     contty    Jun 17 12:49  old   8226
starlan   .         Jun 17 12:50  old   8230
ttymon1   .         Jun 17 12:50  old   8234
ttymon3   .         Jun 17 12:50  old   8235
```

The command

```
who -u
```

lists all users who are currently logged in.

Figure 13-16: who -u **output.**

```
root      console    Jun 17 13:07    .     8303
john      term/32    Jun 17 13:13   0:01   8353
```

If ttymon invokes a service other than login, an entry for this service will appear beginning with a "." and giving the terminal line.

NOTE

To find out which ports are accessible but currently not in use, use the command

```
pmadm -l -t ttymon
```

as described in the section "Managing TTY Ports."

Identifying ttymon Processes

The ps command lists all ttymon processes. Since ttymon port monitors fork a process to handle each connection request, the number of ttymon-related entries that appear in the output of a ps listing may be greater than the number of running ttymon port monitors.

Figure 13-17: Sample output of ps -ef.

```
 UID   PID  PPID  C    STIME TTY       TIME COMMAND
root     0     0  0 17:30:30 ?         0:06 sched
root     1     0  0 17:30:30 ?         1:00 /sbin/init
root     2     0  0 17:30:30 ?         0:01 pageout
root     3     0  0 17:30:30 ?         0:20 bdflush
root     4     0  0 17:30:30 ?         0:00 kmdaemon
root  8215     1  0 12:49:52 ?         0:03 /usr/slan/lib/admdaemon
root  8216     1  0 12:49:49 ?         0:05 /usr/sbin/cron
root  8223     1  0 12:49:51 ?         0:01 /usr/sbin/hdelogger
root  8224     1  0 12:49:51 ?         0:04 /usr/lib/saf/sac -t 300
root  8226     1  0 12:49:49 ?         0:03 /usr/lib/saf/ttymon -g -m ldterm
root  8230  8224  0 12:50:05 ?         0:02 /usr/lib/saf/listen -m slan starlan
root  8234  8224  0 12:50:07 ?         0:04 /usr/lib/saf/ttymon
root  8235  8224  0 12:50:08 ?         0:08 /usr/lib/saf/ttymon
root 12378     1  0 14:26:21 console   0:07 -sh
root   104     1  0 11:08:15 ?         0:09 /usr/sbin/cron
root  8335  8235  0 14:43:34 term/31   0:00 /usr/lib/saf/ttymon
root 12453 12378  0 15:06:36 console   0:01 ps -ef
```

When a ttymon port monitor forks a child to process a connection request (that is, to do baud rate searching, set final termio options, etc., before invoking the service), the port will be identified in the TTY field for this child process (see the example above). For the parent ttymon port monitor process, this TTY field will be empty.

In the above example, there are two ttymon port monitors running, with process IDs 8234 and 8235. Both were started by the SAC. However, the output of the ps command does not identify the port monitors' *pmtags*. The pmtag and process ID of a specific port monitor can be obtained using the who command (see the previous section).

Log Files

Problems often arise when a single port is monitored by more than one process. If a port (for example, /dev/term/11) is used by an enabled service under a ttymon port monitor running under the Service Access Facility, and the same port is also monitored by a ttymon process running in ttymon express mode, (that is, started by init when it reads inittab, not by sac when it reads its administrative file) then the port will behave unpredictably. The system administrator is expected to examine the system for such ambiguously configured ports.

There are also two log files that can be examined for clues to problems related to ttymon port monitors or ports monitored by ttymon port monitors: The Service Access Controller records aberrant port monitor behavior in /var/saf/_log; and each ttymon port monitor has its own log file, /var/saf/*pmtag*/log, where it records messages it receives from the SAC, services it starts, etc.

The command

```
        tail -25 /var/saf/_log
```

will list the most recent 25 entries in the _log file.

Periodically, log files should be cleared or truncated. If you want cron to do the cleanup for you, add the appropriate commands to the file /var/spool/cron/crontabs/root.

Terminal Line Settings

`init` is a general process spawner that is invoked as the last step in the boot procedure. It starts the SAC. The SAC then looks in its administrative file to see which port monitors to start. Each `ttymon` port monitor started by the SAC looks in *its* administrative file for the TTY ports to initialize. For each TTY port initialized, `ttymon` searches the `ttydefs` file for the information it needs to set terminal modes and line speeds. `ttymon` then waits for service requests. When a service request is received, `ttymon` executes the command (usually `login`) associated with the port that received the request. This command is contained in the entry for the port in the port monitor's administrative file.

From the system administrator's point of view, the key elements in managing terminal line settings are the `ttydefs` file and the `sttydefs` command, which maintains the `ttydefs` file.

The `ttydefs` File

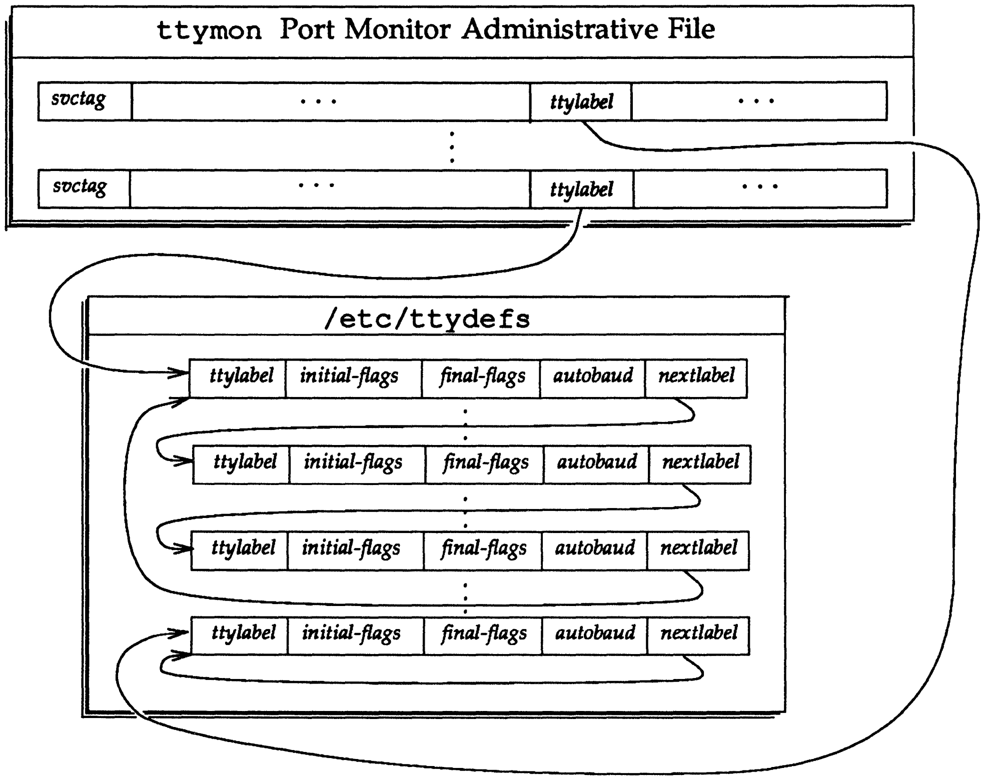
`/etc/ttydefs` is an administrative file used by `ttymon`. It defines speed and terminal settings for TTY ports. The `ttydefs` file contains the information listed below. The figure following shows the relationship between the *ttylabel* and *nextlabel* fields in the `ttymon` administrative files and `ttydefs` files. The figure after that shows a sample `ttydefs` file.

ttylabel	When `ttymon` initializes a port, it searches the `ttydefs` file for the entry that contains the `termio`(7) settings for that port. The correct entry is the one whose *ttylabel* matches the *ttylabel* for the port. The *ttylabel* for the port is part of the *pmspecific* information included in `ttymon`'s administrative file. By convention, *ttylabel* identifies a baud rate (for example, `1200`), but it need not.
initial-flags	Contains the `termio`(7) options to which the terminal is initially set. *initial-flags* must be specified using the syntax recognized by the `stty`(1) command.
final-flags	Contains the `termio`(7) options set by `ttymon` after a connection request has been made and immediately before invoking a port's service. Final flags must be specified using the syntax recognized by `stty`.

autobaud	Autobaud is a line-speed option. When autobaud is used instead of a baud rate setting, `ttymon` determines the line speed of the TTY port by analyzing the first carriage return entered and sets the speed accordingly. If the autobaud field contains the character `A`, the autobaud facility is enabled. Otherwise, autobaud is disabled.
nextlabel	If the user indicates (by sending a BREAK) that the current `ttydefs` entry does not provide a compatible line speed, `ttymon` will search for the `ttydefs` entry whose *ttylabel* matches the *nextlabel* field. `ttymon` will then use that field as its *ttylabel* field. A series of speeds is often linked together in this way into a closed set called a hunt sequence. For example, `4800` may be linked to `1200`, which in turn is linked to `2400`, which is finally linked to `4800`.

All `termio`(7) settings supported by the `stty` command are supported as options in the `ttydefs` file. For example, the system administrator will be able to specify the default erase and kill characters. This was not possible prior to UNIX System V Release 4.

Figure 13-18: Links between the port monitor administrative file and the `ttydefs` file.



NOTE The format of the `/etc/ttydefs` file may change in future releases. For continuity across releases, use the `sttydefs`(1M) command to access this file.

Figure 13-19: Sample `ttydefs` file.

```
# VERSION=1
38400:38400 hupcl erase ^h:38400 sane ixany tab3 hupcl erase ^h::19200
19200:19200 hupcl erase ^h:19200 sane ixany tab3 hupcl erase ^h::9600
9600:9600 hupcl erase ^h:9600 sane ixany tab3 hupcl erase ^h::4800
4800:4800 hupcl erase ^h:4800 sane ixany tab3 hupcl erase ^h::2400
2400:2400 hupcl erase ^h:2400 sane ixany tab3 hupcl erase ^h::1200
1200:1200 hupcl erase ^h:1200 sane ixany tab3 hupcl erase ^h::300
300:300 hupcl erase ^h:300 sane ixany tab3 hupcl erase ^h::19200
```

The `sttydefs` Command

`sttydefs`(1M) is an administrative command that maintains the `ttydefs` file. The `ttydefs` file contains information about line settings and hunt sequences for the system's TTY ports. The `sttydefs` command and the `ttydefs` file together provide the facilities for managing terminal modes and line settings. The `sttydefs` command is used to

- print information contained in `ttydefs`
- add records for terminal ports to the `ttydefs` file
- remove records from the `ttydefs` file

Printing Terminal Line Setting Information

```
/usr/sbin/sttydefs -l [ttylabel]
```

If a *ttylabel* is specified, `sttydefs` prints the `ttydefs` record that corresponds to this *ttylabel*. If no *ttylabel* is specified, `sttydefs` prints this information for all records in the `/etc/ttydefs` file. `sttydefs` verifies that each entry it displays is correct and that the entry's *nextlabel* field refers to an existing *ttylabel*. An error message is printed for each invalid entry detected.

Adding Records to the `ttydefs` File

```
/usr/sbin/sttydefs -a ttylabel [-b] [-n nextlabel] \
   [-i initial-flags] [-f final-flags]
```

`sttydefs` with a `-a` option adds a record to the `ttydefs` file. *ttylabel* identifies the record. The following paragraphs describe the effect of the `-b`, `-n`, `-i`, or `-f` options when used with the `-a` option. The `-a` option is valid only when invoked by a privileged user.

The `-b` option enables autobaud.

The `-n` option specifies the value to be used in the *nextlabel* field. If *nextlabel* is not specified, `sttydefs` will set *nextlabel* to *ttylabel*.

The `-i` option specifies the value to be used in the *initial-flags* field. The argument to this option must be presented in a format recognized by the `stty` command. If *initial-flags* is not specified, `sttydefs` will set *initial-flags* to the `termio`(7) flag `9600`.

The `-f` option specifies the value to be used in the *final-flags* field. The argument to the `-f` option must be presented in a format recognized by the `stty` command. If *final-flags* is not specified, `sttydefs` will set *final-flags* to the `termio`(7) flags `9600` and `sane`.

The following command line creates a new record in `ttydefs`:

```
sttydefs -aNEW -nNEXT -i"1200 hupcl erase ^h" \
   -f"1200 sane ixany hupcl erase ^h echoe"
```

The flag fields shown have the following meanings:

`300-19200`	The baud rate of the line.
`hupcl`	Hang up on close.
`sane`	A composite flag that stands for a set of normal line characteristics.
`ixany`	Allow any character to restart output. If this flag is not specified, only DC1 CTRL-q will restart output.
`tab3`	Send tabs to the terminal as spaces.

`erase ^h`	Set the erase character to CTRL-h. On most terminals a CTRL-h is the backspace.
`echoe`	Echo erase character as the string backspace-space-backspace. On most terminals this will erase the erased character.

Creating a Hunt Sequence

The following sequence of commands adds records with labels `1200`, `2400`, `4800`, and `9600` to the `ttydefs` file and puts them in a circular list or hunt sequence. In the example, the `nextlabel` field of each line is the *ttylabel* of the next line. The *nextlabel* field for the last line shown points back to the first line in the sequence.

The object of a hunt sequence is to link a range of line speeds. Entering a BREAK during the baud rate search causes `ttymon` to step to the next entry in the sequence. The hunt continues until the baud rate of the line matches the speed of the user's terminal.

```
sttydefs -a1200 -n2400 -i 1200 -f "1200 sane"
sttydefs -a2400 -n4800 -i 2400 -f "2400 sane"
sttydefs -a4800 -n9600 -i 4800 -f "4800 sane"
sttydefs -a9600 -n1200 -i 9600 -f "9600 sane"
```

The `ttydefs` file containing these records will look like this:

```
# VERSION=1
1200:1200:1200 sane::2400
2400:2400:2400 sane::4800
4800:4800:4800 sane::9600
9600:9600:9600 sane::1200
```

Removing Records from the `ttydefs` File

```
/usr/sbin/sttydefs -r ttylabel
```

The record for the *ttylabel* specified on the command line is removed from the `ttydefs` file.

The `-r` option is valid only when invoked by a privileged user.

NOTE If a record you remove is part of a hunt sequence, be sure the sequence is repaired. It may be useful to run `sttydefs` with the `-l` option after a record has been removed. `sttydefs -l` will check for incorrect field values and broken hunt sequences and will print error messages.

Setting Terminal Options with the `stty` Command

The `stty(1)` command may be used to set or change terminal options after a user has logged in. A `stty` command line may also be added to a user's `.profile` to set options automatically as part of the `login` process. The following is an example of a simple `stty` command:

```
stty cr0 nl0 echoe -tabs erase ^h
```

The options in the example have the following meanings:

`cr0`	No delay for carriage return or new line. Delays are not used on a video display terminal, but are necessary on some printing terminals to allow time for the mechanical parts of the equipment to move.
`echoe`	Erase characters as you backspace.
`-tabs`	Expand tabs to spaces when printing.
`erase ^h`	Change the character-delete character to CTRL-h. The default character-delete character is the pound sign (#). Most terminals transmit a CTRL-h when the `backspace` key is pressed.

Quick Reference to `ttymon` and Terminal Line Setting

COMMAND SYNTAX	DESCRIPTION
`sacadm -l [-t` *type* `| -p` *pmtag*`]`	Lists all port monitors (`-l` alone), all port monitors of a given type (`-t` *type*), or a single port monitor (`-p` *pmtag*).
`pmadm -l [-t` *type* `| -p` *pmtag*`] [-s` *svctag*`]`	Lists all services for all port monitors (`-l` alone), all services for all port monitors of a given type (`-t` *type*), all services for a specific port monitor (`-p` *pmtag*), or a single service (`-s` *svctag*).
`sacadm -a -p` *pmtag* `-t ttymon -c` *cmd* `-v 'ttyadm -V'`	Adds a `ttymon` port monitor. `ttyadm` used with `sacadm -a` or `pmadm -a` as an argument to the `-v` option provides the comment line containing the `ttymon` version number for the new port monitor administrative file.
`sacadm -r -p` *pmtag*	Removes a port monitor.

`pmadm -a -p` *pmtag* `-s` *svctag* `-i` *id* `[-f ux] -v 'ttyadm -V' \`
 `-m "'ttyadm [-b] [-r` *count*`] [-c] [-h] [-i` *msg*`] \`
 `[-m` *modules*`] [-p` *prompt*`] [-t` *timeout*`] \`
 `-d` *device* `-l` *ttylabel* `-s` *service*`'"`
	Adds a service. `ttyadm` used with `pmadm -a` as an argument to the `-m` option provides the *pmspecific* fields for inclusion in the port monitor's administrative file.

`pmadm -r -p` *pmtag* `-s` *svctag*	Removes a service.
`pmadm -e -p` *pmtag* `-s` *svctag*	Enables a service.

COMMAND SYNTAX	DESCRIPTION
`pmadm -d -p` *pmtag* `-s` *svctag*	Disables the service *svctag*, available through port monitor *pmtag*.
`sacadm -e -p` *pmtag*	Enables all services defined for port monitor *pmtag*.
`sacadm -d -p` *pmtag*	Disables all services defined for port monitor *pmtag*.
`/usr/sbin/sttydefs -a` *ttylabel* `[-b] [-n` *nextlabel*`] \ [-i` *initial-flags*`] [-f` *final-flags*`]`	Adds an entry to the `/etc/ttydefs` file.
`/usr/sbin/sttydefs -l` [*ttylabel*]	Prints terminal line setting information from the `/etc/ttydefs` file for terminal ports with the label *ttylabel*. If no *ttylabel* is specified, prints terminal line setting information for all records in the file.
`/usr/sbin/sttydefs -r` *ttylabel*	Removes records for the *ttylabel* specified from `/etc/ttydefs`.

The Listener

`listen` is a port monitor invoked by the Service Access Controller (SAC). The Service Access Controller is the Service Access Facility's controlling process. It is started by `init` when the system enters multi-user mode. One of the SAC's functions after it is started is to start all port monitors the system administrator has configured.

`listen` monitors a connection-oriented transport network, receiving incoming connection requests, accepting them, and invoking the services that have been requested. The listener may be used with any connection-oriented transport provider that conforms to the Transport Interface (TLI) specification. The Transport Interface is documented in the *Programmer's Guide: Networking Interfaces*. Beginning with UNIX System V Release 4, the listener conforms to the Service Access Facility model.

What `listen` Does

`listen` performs functions common to all port monitors:

- It initializes and monitors `listen` ports and
- it invokes the service associated with a port in response to requests.

The UNIX System V Release 4 listener differs from previous listeners in several ways.

- It allows private addresses for services,
- passes connections (file descriptors) to standing servers,
- supports socket-based services, and
- supports RPC-based services and dynamic addressing.

Private Addresses for Services

Each `listen` service may have a transport address in addition to its service code (*svctag*). This private address is included in the port monitor's administrative file. The inclusion of private addresses for services allows a single `listen` process to monitor multiple addresses. The number of addresses that the listener can listen on is determined by the number of file descriptors available to the process.

Passing Connections to Standing Servers

By default, a new instance of a service is invoked for each connection. Beginning with UNIX System V Release 4, the listener has the ability to pass an incoming connection (file descriptor) to a standing server, eliminating the `fork/exec` overhead for each call (see `nlsadmin`(1M)). This feature is useful for server processes that need to maintain state information.

NOTE A standing server is a server process or service that runs continuously and accepts connections through a FIFO or a named STREAM instead of being `fork`ed and `exec`ed.

Socket-based Services

The listener supports services that use sockets as their interface to the transport provider. Socket-based services are registered with the listener in the same way TLI-based services are, using the Service Access Facility's administrative commands. `listen` supports STREAMS; sockets is implemented as a STREAMS module and a library.

A socket-based service

- may or may not be an RPC service
- may have a statically or dynamically assigned address, or no private address
- may be invoked on each connection or may be a standing server, to which new connections are passed by a FIFO or a named STREAM.

RPC-based Services and Dynamic Addressing

Dynamic addressing is most useful with RPC. RPC transport addresses may be either specified or dynamically assigned. In either case, the listener tells the `rpcbinder` what the address is and monitors it for incoming connections.

In the case of a dynamically assigned address, `listen` asks the transport provider to select a transport address each time the listener begins listening on behalf of the service.

When service addresses are dynamically assigned, the assigned address is written to the listener's log file.

`listen` and the Service Access Facility

The Service Access Facility (SAF) provides a generic interface to which all port monitors must conform. `listen` is a port monitor under the Service Access Facility's controller, the Service Access Controller. (See "Overview of the Service Access Facility," "Port Monitor Management," and "Service Management," above, for a description of the Service Access Facility, the administrative files it maintains, and the commands used for port monitor and service administration.)

There can be multiple invocations of `listen` port monitors, each identified by a unique *pmtag*. Each of these port monitors can monitor multiple ports for incoming connection requests.

A port has one and only one service associated with it. Each port, and its associated service, is identified by a service tag, *svctag*. Service tags for any given port monitor are unique.

When the Service Access Controller starts a port monitor, the port monitor reads its administrative file, which contains information about which ports to monitor and what service (i.e., process) is associated with each port.

The `nlsadmin` Command

The Service Access Facility requires each type of port monitor to provide an administrative command. This command must format information derived from command-line options so that it is suitable for inclusion in the administrative files for that port monitor type. The command may also perform other port monitor-specific functions.

`nlsadmin` is the listener's administrative command. The `nlsadmin` command formats information based on the options with which it is invoked and writes this information to the standard output.

`nlsadmin` is one of the arguments pmadm uses with the -a option to format information in a way suitable for inclusion in a `listen` administrative file. `nlsadmin` presents this information (as standard output) to pmadm, which places it in the file. This use of `nlsadmin` is described below under "Adding a `listen` Port Monitor." *pmspecific* information in a port monitor administrative file will be different for different port monitor types.

`nlsadmin` is also included on the `sacadm` command line when a port monitor is added to the system. It is used to supply the `listen` version number for inclusion in a port monitor's administrative file.

NOTE The port monitor administrative file is updated by the Service Access Controller's administrative commands, `sacadm` and `pmadm`. `nlsadmin` merely provides a means of presenting formatted port monitor-specific (i.e., `listen`-specific) data to these commands.

The `sacadm` command line uses `nlsadmin` only with the `-V` option. `nlsadmin -V` tells the SAC the version number of the `listen` command being used.

Earlier versions of `nlsadmin` allowed the system administrator to add and delete services, start and stop the listener, and query the status of services using the `nlsadmin` command directly. Although this use of `nlsadmin` is retained for compatibility, these functions are now provided by the SAF administrative commands. This use of the SAF commands is described in the sections that follow. All uses of `nlsadmin`, including its use with the SAF administrative commands, are described on the `nlsadmin`(1M) manual page.

Under the SAF, it is possible to have multiple instances of the listener on a single *net_spec*. A new option, `-N` *pmtag*, can be used in place of the *net_spec* argument. This argument specifies the tag by which an instance of the listener is identified by the SAF. If the `-N` option is not specified (i.e. the *net_spec* is specified in the invocation), then it will be assumed that the last component of the *net_spec* represents the tag of the listener for which the operation is destined.

Managing `listen` Ports

Finding Out What `listen` Port Monitors Are Configured

```
sacadm -l [ -t listen ]
```

The `sacadm` command with only a `-l` option lists all port monitors currently defined for the system. For example:

```
PMTAG     PMTYPE  FLGS  RCNT   STATUS        COMMAND
starlan   listen  dx    5      NOTRUNNING    /usr/lib/saf/listen -m slan starlan  #
ttymon1   ttymon  -     0      ENABLED       /usr/lib/saf/ttymon                  #
ttymon2   ttymon  -     0      ENABLED       /usr/lib/saf/ttymon                  #
```

Finding Out What Services Are Configured for a `listen` Port Monitor

pmadm -l [-p *net_spec*] [-s *svctag*]

pmadm with only a -l will list all services for all port monitors on the system. If a port monitor is specified (-p), all services for that port monitor will be listed. The following is a sample listing for the command

```
pmadm -l -p starlan
```

```
PMTAG   PMTYPE  SVCTAG FLGS ID      PMSPECIFIC
starlan listen  101    -    listen  - c - /usr/lib/uucp/uucico -r0 -unuucp -iTLI \
                                          #UUCP access direct to server
starlan listen  102    -    listen  - c - /usr/slan/lib/ttysrv -d -n ntty,tirdwr,ld0 \
                                          #UUCP access to server via login
starlan listen  1000   -    listen  - c - /usr/slan/lib/tstserver #TP TEST SERVER
starlan listen  105    -    listen  - c - /usr/net/servers/rfs/rfsetup #RFS SERVER
starlan listen  0      -    root    - c - sfbul.serve c - /usr/lib/saf/nlps_server #NLPS server
starlan listen  1      -    listen  - c - sfbul c - /usr/slan/lib/ttysrv -m ntty,tirdwr,ld0 \
                                          #TTY SERVER
```

The following command lines list the addresses associated with general `listen` service (0) or with login service (1).

pmadm -l -p *net_spec* -s 0
pmadm -l -p *net_spec* -s 1

NOTE By definition, service code 0 is for the `nlps_server`, which is a service that provides compatibility with pre-UNIX System V Release 4 `listen` service requests. Service code `1` is for remote login (i.e., `cu` over a network).

Adding a `listen` Port Monitor

```
sacadm -a -p pmtag -t type -c cmd -v `pmspecific -V` \
   -n count [ -f dx ] [ -z script ] [ -y comment ]
```

The following example shows how the listener's administrative command, `nlsadmin`, can be used to obtain the current version number of the listener's administrative file when used with `sacadm` to add a `listen` port monitor.

```
sacadm -a -p starlan -t listen \
       -c "/usr/lib/saf/listen -m slan" \
       -v `nlsadmin -V`
```

This command line adds a line to the SAC's administrative file. The options that may be used with `sacadm -a` are described under "Port Monitor Management," above, and in the `sacadm`(1M) and `nlsadmin`(1M) manual pages.

NOTE If the port monitor being added has the same name as an existing port monitor, the system administrator must remove the old one before adding the new one.

Removing a `listen` Port Monitor

```
sacadm -r -p net_spec
```

For example,

```
sacadm -r -p starlan
```

The SAC removes the line for port monitor `starlan` from its administrative file. The port monitor directory will remain in `/etc/saf` but will be removed and recreated when a new port monitor with the same name is added. To make changes to a port monitor entry, always remove the entry and add a new entry using the `sacadm` command. Do not edit the SAC administrative file.

Adding a Service

```
pmadm -a -p net_spec | pmtag -s svctag -i id \
-m "`nlsadmin options`" -v `nlsadmin -V` -y comment
```

The following example adds a new service, `/usr/bin/cmd`, to a listener with *pmtag* `listen` running as a port monitor under the Service Access Facility. The new service has service tag `23`, identity `guest`, and no private address:

```
pmadm -a -p listen -s 23 -i guest \
        -m `/usr/sbin/nlsadmin -c /usr/bin/cmd` \
        -v `/usr/sbin/nlsadmin -V`
```

CAUTION The same address cannot be monitored by more than one `listen` port monitor at any given time. The first attempt to listen on an address will bind successfully; subsequent attempts will fail to bind. If both static and dynamic addresses are monitored by more than one listener, the static addresses are bound first, then the dynamic addresses. Mixing multiple listeners—each of which has static and dynamic addresses specified—may result in unpredictable behavior.

See "Adding a Service" under "Service Management," or the pmadm(1M) and `nlsadmin`(1M) manual pages for a full description of the pmadm command line options.

Removing a Service

```
pmadm -r -p net_spec | pmtag -s svctag
```

For example,

```
pmadm -r -p starlan -s 23
```

removes service `23` from the `starlan` listener.

Enabling and Disabling Services

```
pmadm -e -p net_spec -s svctag
pmadm -d -p net_spec -s svctag
```

To enable a service on a specific port, first find out which port monitor is monitoring the port. Enter

```
pmadm -l -t listen
```

This lists all services defined for listen-type ports.

If the port monitor is `starlan` and the service tag is `101`, the command

```
pmadm -e -p starlan -s 101
```

will enable service `101`.

To verify that the port has been enabled, enter

```
pmadm -l -p starlan -s 101
```

The `x` will have been removed from the FLGS column in the entry for this service.

When a service is disabled, all subsequent connection requests for the service will be denied. Using the same example,

```
pmadm -d -p starlan -s 101
```

will restore the `x` to the FLGS field in the entry for service 23.

Disabling All Services Monitored by a `listen` Port Monitor

sacadm -d -p *pmtag*

To disable all services defined for the port monitor `starlan`, enter

```
sacadm -d -p starlan
```

Any future connection requests for services managed by this port monitor will be denied until the port monitor is enabled.

The command

```
sacadm -e -p starlan
```

will re-enable port monitor `starlan`.

Configuration Files

As a port monitor under the Service Access Facility, `listen` can customize the environment of each service it starts. It does this by interpreting a per-service configuration script, if one exists, immediately before starting the service. Per-service configuration scripts are optional. Configuration scripts are installed by the system administrator, using the `pmadm` command with `-g` and `-z` options (see the `pmadm`(1M) manual page).

It is also possible to customize the environment of a `listen` port monitor. A per-port monitor configuration script is defined using the `sacadm` command with `-g` and `-z` options (see the `sacadm`(1M) manual page). The environment modifications made by a port-monitor configuration script are inherited by the port monitor and all the services it invokes. The environment of any particular service can then be customized further by using a per-service configuration script.

The `doconfig`(3N) manual page describes the language in which configuration scripts are written.

Configuration scripts are not normally needed for basic operations.

Log Files

The listener creates and manages the log files `/var/saf/`*pmtag*`/log` and `/var/saf/`*pmtag*`/o.log`. Log file entries are in the following format:

date time; PID; message

date and *time* show when the entry was made. *PID* is the ID of the process that made the log entry. *message* gives a description of the event or error that caused the log message.

The following events are logged:

- each connection that arrives
- each service that is started
- each file descriptor passed over a pipe

- state changes that occur
- errors and unusual conditions

The log files are held open by the listener process. Entries are made by two types of processes: the listener process (`listen`) and the NLPS server process (`nlps_server`). `nlps_server` is a service that provides compatibility with pre-UNIX System V Release 4 service requests.

Quick Reference to `listen`

COMMAND SYNTAX	DESCRIPTION
`sacadm -l [ -t` *type* `]`	Lists status information for all port monitors (`-l` alone) or for all port monitors of a given type (`-t` *type*).
`pmadm -l -p` *net_spec* `[ -s` *svctag* `]`	If *svctag* is supplied, lists status information for the service. If no service is specified, lists status information for all services under *net_spec*.

`sacadm -a -p` *net_spec* `|` *pmtag* `-t listen \`
 `-c "/usr/lib/saf/listen` *net_spec*`" \`
 `-v 'nlsadmin -V'`
	Adds a `listen` port monitor.

`sacadm -r -p` *net_spec*	Removes a `listen` port monitor.
`pmadm -a -p` *net_spec* `|` *pmtag* `-s` *svctag* `-i` *id* `\` `-m "'nlsadmin` *options*`'" -v 'nlsadmin -V' -y` *comment*	Adds a service under a `listen` port monitor.
`pmadm -r -p` *net_spec* `|` *pmtag* `-s` *svctag*	Removes a service under a `listen` port monitor.
`pmadm -e -p` *net_spec* `-s` *svctag*	Enables service *svctag* under port monitor *net_spec*.
`pmadm -d -p` *net_spec* `-s` *svctag*	Disables service *svctag* under port monitor *net_spec*.
`sacadm -d -p` *net_spec*	Disables all services under port monitor *net_spec*.
`sacadm -e -p` *net_spec*	Enables all services under *net_spec*.

14 Software Management

Introduction

This chapter tells you how to install software packages on your system, how to store packages on your system for installation later, and how to remove packages. You can do any of these functions by selecting the appropriate task from a series of menus provided for administration. To access the system administration menu for installing and removing software, type

```
sysadm software
```

The following menu will appear on your screen:

```
1        Software Installation and Information Management

check        - Checks Accuracy of Installation
defaults     - Sets Installation Defaults
install      - Installs Software Package
interact     - Stores Interactions with Package
list         - Display Information about Packages
read_in      - Stores Packages Without Installing
remove       - Removes Packages
```

If you prefer not to use the menus, you can perform the same tasks by executing shell-level commands, instead. The following table shows the shell commands that correspond to the tasks listed on the menu.

Task to Be Performed	`sysadm` Task	Shell Command
Set installation defaults	`defaults`	`vi`(1) `admin`(4)
Store interaction with packages	`interact`	`pkgask`(1)
Install software package	`install`	`pkgadd`(1)
Check accuracy of installation	`check`	`pkgchk`(1)
Show installed packages	`list`	`pkginfo`(1)
Store packages (not install them)	`read_in`	`pkgadd`(1)
Remove packages	`remove`	`pkgrm`(1)

Each task is explained fully later in this chapter. Details about the admin file and all commands except `vi` are available in the *System Administrator's Reference Manual*. For details about `vi`, see the *User's Reference Manual*.

An Overview of Software Management

Software management responsibilities entail setting up your software installation environment, installing software on your machine, keeping track of it while it is on your system, and removing it as necessary. This chapter provides the information necessary for performing these responsibilities and detailed instructions for the following tasks:

- Setting installation defaults

  Describes how to create and use the admin file, which defines values for your installation default parameters.

- Storing interactions with a package

  Describes how to use the `pkgask` command to create a file used when installing software in what is known as non-interactive mode.

- Installing software packages

  Describes the installation command `pkgadd`, how it works, and how to use its various options. Covers how to execute both interactive and non-interactive installation and also contains information on common installation errors.

- Installing software from a remote machine

  Provides an example of remote installation. The example shows how to use the Remote File Sharing utilities, which make remote file names appear to be local.

- Checking installation accuracy

  Describes how to use the `pkgchk` command to check the integrity of your packages after they have been installed.

- Showing information about installed packages

  Describes the `pkginfo` command and the various types of displays you can create to show information about packages installed on your system.

- Storing packages without installing them

  Describes how to spool a package onto your machine for future installation.

- Removing packages

  Describes how to remove a package with the pkgrm command.

The following section defines some basic terms you will need to understand to install software packages properly. How to name packages on the command line, what a relocatable package is, and the installation software database are described, along with the differences between interactive and non-interactive installation.

Basic Software Management Terminology

A software package consists of components (for example, compiled programs, files, installation scripts) that are delivered on an installation medium. An installation medium is any physical storage medium, such as a diskette or tape, on which packages can be stored. Software packages are installed from an installation medium to your machine by using the pkgadd command.

There are two formats for packages on a medium: a datastream or a filesystem. The datastream format consists of a header and a series of cpio archives and it can be read from any raw device. pkgadd automatically determines the format of the medium when it reads the first volume.

Software packages can also be installed from a directory. This is quite useful in remote file sharing environments where a directory containing packages is advertised from a server machine to a large network.

You can install software packages in one of two modes: interactive and non-interactive. The interactive mode allows pkgadd to query you when problems occur during installation and receive input on how to proceed. In the *non-interactive mode* pkgadd does not query you; instead you give pkgadd the information it needs by supplying two files: an admin file and a response file.

- The admin file supplies pkgadd with instructions on what actions to take when it encounters certain situations during installation. These instructions are defined in the form of installation default parameters. The values assigned to these parameters define the installation defaults, which are the actions pkgadd will take when it encounters the situation associated with that parameter.

- The response file, created by running pkgask, holds answers to questions the software package would ask you during interactive installation. Because no interaction can occur during non-interactive installation, the package reads the response file to receive its information.

If pkgadd encounters a problem and cannot find instructions in the admin file or response file, it terminates the installation.

You can also spool a package instead of installing it. This causes the contents of a package to be copied from an installation medium into a spool directory. No other type of installation action is taken. (For example, the installation scripts are not executed.) A spooled package can be installed later from the spool directory.

Information for all packages installed on a system is kept in the installation software database. There is an entry for every component in a package, with information such as the component name, where the component resides, and the component type. Entries are added and removed automatically by pkgadd and pkgrm. You can view the information in the database by using the pkginfo command.

Two types of information are associated with each package component. The *attribute information* describes the component itself. For example, the component's access permissions, owner ID, and group ID are attribute information. The content information describes the contents of the component, such as file size and time of last modification.

The installation software database keeps track of the package status. A package can be either fully installed, meaning it has successfully completed the installation process, or partially installed, meaning that it did not successfully complete the installation process. In this case, portions of a package may have been installed before installation was terminated; thus, part of the package is installed, and recorded in the database, and part is not. When you reinstall the package, you are prompted to start at the point where installation stopped because pkgadd can access the database and know what has already been installed. You can also remove the portions that were installed based on the information stored in the installation software database.

Variations of a software package can differ by version or architecture or both. Multiple variations of the same package can reside simultaneously on the same machine. Each variation is known as a package instance. pkgadd assigns a package identifier to each package instance at the time of installation. The package identifier is the package abbreviation with a numerical suffix. This identifier distinguishes an instance from any other package, including other instances of the same package.

A package is delivered either with a fixed location, meaning that its location on a machine is defined by the package itself and cannot be changed, or it is delivered as relocatable, meaning it does not require installation in a specific place on a machine. If a package is relocatable, you will be asked during installation where you want to install it.

How to Name Packages in a Command

Multiple variations of a software package may reside on your system simultaneously because, for example, there may be a number of releases that need to coreside. Each variation of a package is known as an "instance" and is treated as a separate entity. Three parameters, defined by the package developer, combine to uniquely identify each entity. You cannot install on the same machine two instances of a package that have identical values for all three parameters. These parameters are:

- `PKG`, which defines the software package abbreviation (this remains constant for every instance of a package)
- `VERSION`, which defines the software package version
- `ARCH`, which defines the software package architecture

For example, two different versions of a package that run on the same machine might be identified as:

```
PKG="abbr"                PKG="abbr"
VERSION="release 1"       VERSION="release 2"
ARCH="3B2"                ARCH="3B2"
```

At the time of installation, `pkgadd` assigns a numerical suffix to the package abbreviation. The combination, for example `mypkg.2`, is known as the package identifier. This ID maps the three pieces of information that identify a package instance to one name, which becomes the name of this instance on your machine.

The first instance of a package installed on a system does not have a suffix, and so its package identifier will be the package abbreviation. pkgadd assigns subsequent instances a suffix, beginning with .2. An instance is given the lowest extension available and so may not correspond to the order in which a package was installed. For example, if mypkg.2 was deleted after mypkg.3 was installed, the next instance to be added would be named mypkg.2.

When asked for *pkgid* in any of the procedures described in this chapter, you must use the package identifier. Remember that when you have only one instance of a package on a machine, which is probably the most common situation, the package identifier is the package abbreviation.

To execute a single command for all instances of a package, specify the abbreviation of the desired package, followed by .*. For example, if you want to display information about all instances of a package with the abbreviated name mypkg, run the pkginfo command as follows:

```
pkginfo mypkg.*
```

If you want to display information about only one instance of this package, such as the third instance of mypkg, you must specify both the package abbreviation and the number of the desired instance, as follows:

```
pkginfo mypkg.3
```

NOTE You can use the pkginfo command to look up the package identifier of a package instance. See "Showing Information About Installed Packages" for details.

Relocatable Packages

Some packages require that their objects be installed in a particular directory, but for many packages this is not true. It is possible that only parts of a package must reside in a specific place or that no objects must adhere to such a requirement. The package developer can specify that files may be placed in arbitrary locations — that is, that files are relocatable. In turn, the pkgadd command allows you to specify where these relocatable files should be installed on your machine.

The directory portion of a relocatable object pathname is filled in as the package is installed. There are four places from which the directory value can originate:

- The package itself may ask you where you want to install its relocatable objects.
- The `pkgadd` command asks you where you want to install a relocatable package, if there is no default set in the admin file and if you are operating in interactive mode.
- The admin file can assign a value to an installation default parameter that defines the directory where relocatable packages should be installed. If set, the value of this parameter is used as the directory portion of the name.
- The package also delivers a directory name where relocatable objects should be installed if there is a question as to where they should be placed — for example, if you have no default defined in the admin file and are operating in non-interactive mode.

At the time of installation, all relocatable objects are given full pathnames and any variable value associated with them is resolved.

Interactive and Non-Interactive Modes of Installation

The software installation command, `pkgadd`, performs numerous checks during installation. For example, it checks for available disk space and setuid processes. `pkgadd` asks for your input interactively if it identifies a potential problem, such as not enough disk space, and the admin file did not specify how to handle it.

The `pkgadd` command can be run in non-interactive mode by using the `-n` option; however, the same checks are performed, regardless of the mode. This means that in non-interactive mode the command might still need information on how to handle a potential problem; `pkgadd` checks two files when invoked in non-interactive mode for this information. These files are the admin file, which defines default actions for problems, and the response file, which defines answers to questions the package would ask in interactive mode. If you choose to install a package without interaction, you need to determine whether either or both of these files are needed.

Four sections in this chapter should prepare you for installing a package in non-interactive mode:

- "Installing a Package in Non-interactive Mode"
- "Setting Installation Defaults" (defines the admin file and tells how to create one)
- "Storing Interactions with a Package" (defines the response file and tells how to create one)
- "Suggestions for Installing Your Software" (this section contains a sequence for non-interactive installation)

The Installation Software Database

The installation software database stores information about all packages installed on the system. There are entries for every component of a package. An entry contains a record of the package to which a component belongs; other packages that might reference the component; and information such as pathname, where the component resides and the component type.

The system uses this database to see if an object is shared by more than one package, to see if other packages depend on it, or to perform a number of other checks when adding or removing a package. When a package is installed or removed, information about it is automatically added to or removed from this database.

You can use the `pkginfo` command to survey the contents of the installation software database. The commands `installf` and `removef` can be used to modify its contents. See the *System Administrator's Reference Manual* for details on these two commands.

Suggestions for Installing Your Software

When you have a software package you want to install, pkgadd allows you three choices for putting that software on your machine:

- You can install a package in interactive mode.

  This choice lets you take full advantage of the sophistication of the pkgadd command. In interactive mode, it can inform you of potential problems in the installation as they occur. You can then instruct pkgadd how to proceed.

- You can install a package in non-interactive mode.

  This choice does not let pkgadd interact with you during the installation process. It can be faster than the interactive mode, but you will not be able to supply instructions as problems arise. In some cases, such as background or batch execution, interaction would not be possible. In such a case, using the non-interactive mode would be mandatory.

- You can copy a package to a spool directory.

  Spooling a package copies a software package from the installation medium (without installing it) and places it in a directory on your machine. Such a package can then be installed later from that directory.

Preparing for Installation

Before installing any packages, you should consider the values for your installation default parameters. These are the parameters pkgadd uses for instructions when problems are encountered in both interactive and non-interactive installation. To set up installation defaults, use the `defaults` interface task or use an editor to create an admin file. Refer to "Setting Installation Defaults" for detailed information on what default parameters exist and the values they can be assigned.

It is not necessary to create a new admin file for every installation. A generic admin file is delivered with your UNIX System V Release 4 software. All the defaults defined in this file request that you be queried if a problem occurs. If these defaults are acceptable in all of your package installations, you do not need to create any new admin files. You can create multiple admin files, if you want a number of default sets, for different installation situations.

NOTE Do not make changes to the system-supplied admin file named `default`. If you want to define different parameter values, create a new admin file.

When you invoke the `pkgadd` command, it automatically uses the system supplied file. If you want to use an admin file that you created, you must supply the file name after the `-a` option.

Interactive Installation Checklist

When installing a package in interactive mode, you should:

1. Decide how your installation defaults should be defined for this installation. Choose an admin file that establishes these defaults. If necessary, create a new admin file by using an editor (or by using the `defaults` task from the `sysadm` menu.)
2. Install the package by running the `pkgadd` command (or by selecting the `install` task from the `sysadm` menu).

Non-Interactive Installation Checklist

When installing a package in non-interactive mode, you should:

1. Decide how your installation defaults should be defined for this installation. Choose an admin file that establishes these defaults, or if necessary, create a new admin file using the `defaults` interface task or an editor. This step is more important when installing in non-interactive mode because the system will not be able to question you if problems arise.
2. Determine whether you need to create a response file. Some packages include scripts that will ask for input during installation. If so, that information is necessary even when installation is in non-interactive mode. To supply the package scripts with this information, you must answer the questions before installation and store those answers in a response file. The response file supplies the information to the package scripts. To create a response file, run the `pkgask` shell command (or select the `interact` task from the `sysadm` menu).

NOTE To find out if a package is interactive, and thus needs a response file for non-interactive installation, run the `pkgask` command. It will either begin creating a response file or inform you that a response file is not needed.

3. Install the package by running the `pkgadd` command (or by selecting the `install` task from the `sysadm` menu). You need to use the `-n` option to ask for non-interactive mode, the `-r` option to supply a response file name, and the `-a` option to supply an admin file name.

Spooling Checklist

When spooling a package instead of installing it, you should:

1. Run the `pkgadd` command with the `-s` option (or select the `read_in` task from the `sysadm` menu).
2. Follow the suggested sequences outlined above and name the spool directory as the installation medium, when you are ready to install the spooled package.

Setting Installation Defaults

The commands you use to install and remove packages, pkgadd and pkgrm respectively, check for errors during execution. When a problem occurs, these commands check the admin file for instructions on how to proceed. This file defines values for eleven parameters, each parameter assignment supplying a resolution to a potential problem.

UNIX System V Release 4 provides a default admin file named `default`, located in the `/var/sadm/install/admin` directory. If you want to assign different values for your installation default parameters, you can create your own admin file. As a matter of fact, you can create and use multiple admin files.

NOTE Do not make changes to the system-supplied admin file named `default`. If you wish to define different parameter values, create a new admin file.

pkgadd and pkgrm use `default` unless given the name of an alternative admin file. Using the -a option of the pkgadd or the pkgrm command lets you name an alternative file.

pkgadd and pkgrm automatically look in the `/var/sadm/install/admin` directory for any file named after the -a option unless you give a full pathname. It is best to place any admin file you create in this directory.

Creating an Admin File

Create an admin file with the editor of your choice. You can use any name for a new admin file. Define each parameter on a single line in the following format:

param=value

pkgadd looks in the `/var/sadm/install/admin` directory for admin files. If you put your admin file in this directory, you only need to supply the file name after the -a option when executing pkgadd. You can create your admin files in other directories, but if you do, you must supply the full pathname after the -a.

Eleven parameters can be defined in this file. A description of each, along with a list of permissible values, are shown in the list that begins on the next page. Any of these parameters may be assigned the value ask, which means that if the situation occurs you will be asked to give instructions at that time.

An admin file is not required to assign values to all eleven parameters. However, if pkgadd needs a parameter value and one is not assigned, the default value `ask` is used.

NOTE: The value `ask` cannot be defined in an admin file that will be used for non-interactive installation. Doing so causes installation to fail when a problem occurs.

- `basedir`

  Indicates the base directory where relocatable packages will be installed. The parameter may contain `$PKGINST` to indicate a base directory that is to be a function of the package instance.

  For example, `basedir=/opt/$PKGINST` will place all relocatable packages that use the basedir parameter in a sub-directory of `/opt` with the same name as the package instance.

- `mail`

  Defines a space-separated list of users to whom mail should be sent following installation of a package. If the list is empty, no mail is sent. If the parameter is not present in the admin file, the default value of `root` is used. (The value `ask` cannot be assigned to this parameter.)

- `runlevel`

  Indicates resolution if the machine run level is not correct for the installation or removal of a package. Options are:

  - `nocheck`: do not check for run level
  - `quit`: abort installation if run level is not met

- `conflict`

  Indicates resolution if installation will cause a previously installed file to be overwritten, modified, or have its permissions changed, thereby creating a possible conflict between packages. Options are:

  - `nocheck`: do not check for conflict; files in conflict will be overwritten

    - quit: abort installation if conflict is detected
    - nochange: override installation of conflicting files; they will not be installed

- setuid

  Checks for executables that will have setuid or setgid bits enabled after installation. Options are:

    - nocheck: do not check for setuid executables
    - quit: abort installation if setuid processes are detected
    - nochange: override installation of setuid processes; processes will be installed without setuid bits enabled

- action

  Determines whether installation scripts provided by package developers may contain a possible security impact (for example, by enabling the setuid or setgid bits). Options are:

    - nocheck: ignore security impact of scripts
    - quit: abort installation if scripts may have a negative security impact

- partial

  Checks to see if a version of the package is already partially installed on the system. Options are:

    - nocheck: do not check for a partially installed package
    - quit: abort installation if a partially installed package exists

- instance

  Determines how to handle installation if a previous instance of the package (including a partially installed instance) is already installed on the system. Options are:

- quit: exit without installing if an instance of the package already exists (does not overwrite existing packages)
- overwrite: overwrite an existing package if only one instance exists. If there is more than one instance, but only one has the same architecture, it overwrites that instance. Otherwise, the administrator is prompted with existing instances and asked which to overwrite. (If installed in non-interactive mode, installation terminates.)
- unique: do not overwrite an existing instance of a package. Instead, a new instance of the package is created. The new instance will be assigned the next available package identifier.

■ idepend

Controls resolution during package installation if package dependencies are not met. Options are:

- nocheck: do not check package dependencies
- quit: abort installation if package dependencies are not met

■ rdepend

Controls resolution during package removal if other packages depend on the one to be removed. Options are:

- nocheck: do not check package dependencies
- quit: abort removal if package dependencies are not met

■ space

Controls resolution if disk space requirements for package are not met. Options are:

- nocheck: do not check space requirements (installation fails if it runs out of space)
- quit: abort installation if space requirements are not met

The default admin file (/var/sadm/install/admin/default) defines mail as root and all other parameters as ask. The sample admin file shown below defines parameters differently than default.

Figure 14-1: Sample admin file

```
basedir=default
runlevel=quit
conflict=quit
setuid=quit
action=quit
partial=quit
instance=unique
idepend=quit
rdepend=quit
space=quit
```

Using `sysadm` to Create an Admin File

When you create an admin file with `sysadm`, a series of questions is presented to you. `sysadm` determines the value that should be assigned to each parameter based on your responses. If you are uncertain as to the meaning of the installation parameters and how their values will affect installation, use `sysadm` to help you create your admin files by typing

```
    sysadm software
```

and then selecting `defaults` from the `Software Installation and Information Management` menu.

An admin file created with `sysadm` is automatically placed in the default directory `/var/sadm/install/admin`.

Installing a Package with an Alternative Admin File

Unless you specify differently, pkgadd uses the admin file default. To inform pkgadd that you want to use an alternative admin file, invoke it with the -a option. For example, to install a package named mypkg using an admin file named myadmin, execute the following:

```
pkgadd -d diskettel -a myadmin mypkg
```

Unless you specify a full pathname after the -a, pkgadd looks for the admin file in the /var/sadm/install/admin directory.

NOTE This example shows the use of a device alias, diskettel, in place of a device pathname, such as /dev/diskette. The device alias is defined in the device database (/etc/device.tab), as described in the "Storage Device Management" chapter.

Removing a Package with an Alternative Admin File

Two of the parameters set in the admin file affect package removal. They are runlevel (which gives instructions for what to do when the run level is not met) and rdepend (which defines whether or not to check for other packages with dependencies on this package). You can have alternative admin files to assign different values to these two parameters.

pkgrm uses the admin file default unless you use the -a option to inform pkgrm that you want to use an alternative admin file. For example, to remove a package named mypkg using an admin file named myadmin, execute the following:

```
pkgrm -a myadmin mypkg
```

Unless you specify a full pathname after the -a, pkgadd looks for the admin file in the /var/sadm/install/admin directory.

Storing Interactions with a Package

Before you install a package in non-interactive mode, you must prepare answers for the questions that a package installation script would ask you during the installation process. The `pkgask` command executes the appropriate installation script, thus showing you the questions and allowing you to respond to them. Your answers are stored in a file, called a response file.

You supply a name for the response file on the command line when you execute `pkgadd` to install the package in non-interactive mode. The installation script will use the response file to access the information as it is necessary.

Creating a Response File

To create a response file, execute

```
pkgask  -r  response pkgid
```

where *response* is the full pathname of the file in which your responses will be saved and *pkgid* is the package identifier for the package whose interactions you are storing. (Use `pkginfo` to find out the package identifier if you do not know it. See "Showing Information About Installed Packages" for details on `pkginfo`). If *response* is a directory, a file named `response/pkgid` is created.

NOTE You must use a package identifier with a numerical suffix if multiple versions reside on the installation medium. When there is only one version of a package on a medium, the package identifier is the package abbreviation without a suffix.

The package identifier suffix defines the package instance only on that particular medium. A new suffix will be assigned when this package is installed on your system. To find out what instances are available on a medium, run `pkginfo -d` *device*.

Using the Response File When Installing a Package

After storing your response to the package installation script, use the response file as input to the pkgadd command by executing a command line such as the following:

```
pkgadd -d device -n -r response pkgid
```

where *device* is the name of the device (or directory) on which the package currently resides, −n requests non-interactive installation, *response* is the file you created with the pkgask command, and *pkgid* is the package identifier of the package to be installed.

Using the example above, which names a specific response file, you can install only one package at a time. If you use the −r option to specify the name of a directory that holds several response files corresponding to the packages you wish to install, you can install several packages with a single command. The command line format in such as case is as follows:

```
pkgadd -d device -n -r response_dir pkgA pkgB pkgC
```

Installing Software Packages

Software package installation is the process of copying a software package from the installation medium (such as, a diskette) onto your machine, performing any actions requested by installation scripts, and recording package objects in the installation software database. Installation can be performed in either interactive or non-interactive mode.

Installing a Package in Interactive Mode

The default installation mode is interactive. To interactively install a software package named pkgA from a floppy diskette named diskette1, you would enter:

```
pkgadd -d diskette1 pkgA
```

NOTE This example shows the use of a device alias, diskette1, in place of a device pathname, such as /dev/diskette. The device alias is defined in the device database (/etc/device.tab), as described in the "Storage Device Management" chapter.

You can install multiple packages at one time, as long as you separate package names with white space, as follows:

```
pkgadd -d diskette1 pkgA pkgB pkgC
```

If you do not name the device on which the package resides, the command checks the default spool directory (/var/spool/pkg). If the package is not there, installation fails. The name given after the -d option must be a full pathname to a device or directory or the device alias (as shown in the example).

NOTE You must use a package identifier if multiple versions co-reside on the installation medium. In most cases, there will only be one instance of a package on a medium and the package identifier will be the package abbreviation without a suffix.

Be aware that the suffix of a package identifier defines the package instance on that particular medium. A new package identifier will be assigned this package when it has been installed on your system. (Use pkginfo -d *device* to find out what instances are on a medium.)

pkgadd is a sophisticated command that will always interact with you if it has a question about what it should do. For example, if you supply a device name, but not a package name, it will show you a list of packages on the device (or directory) specified and ask you to choose one to install. In addition, there are a number of potential problems for which it checks. If it finds a problem and does not know what action to take, you will be prompted to supply it with the information it needs.

Interacting with pkgadd

When pkgadd encounters a problem, it first checks the admin file for instructions. If no instructions exist, or if the instructions specify to consult the administrator (meaning a parameter is defined as ask), pkgadd gives you a message describing the problem and prompts you for a reply. The prompt is usually `Do you want to continue with this installation.` You should respond with yes, no, or quit. If you have given more than one package, as in the second example shown previously, no stops installation of the package being installed but informs pkgadd to continue with installation of the other packages. quit tells pkgadd to stop installation of all packages.

Refer to the section "Setting Installation Defaults" for a description of the potential problems for which pkgadd checks and the default instructions you can define for each problem.

Installing a Package in Non-interactive Mode

The installation described thus far has been in the interactive mode. You can install a package and request that no interaction between you and the pkgadd command take place. This is known as the non-interactive mode. To install a package in non-interactive mode, use the -n option. Entering

```
pkgadd -d diskette1 -n mypkg
```

installs mypkg without interaction.

In some instances, non-interactive installation of a package will require a response file. This file holds answers to questions that the software package would ask you during interactive installation. Since no interaction can occur during non-interactive installation, the package will read the response file to receive its information. You create a response file using the pkgask command. Instructions for creating a response file are given in the procedure entitled "Storing Interactions with a Package."

To install mypkg in non-interactive mode and supply a response file named myresponse, enter

```
pkgadd -d diskettel -n -r /pkgA/myresponse mypkg
```

The pathname given after the -r option must be a full pathname.

NOTE You may also want to use an admin file other than default when installing a package in non-interactive mode. To do so, you must create an admin file and then use the -a option of pkgadd. Procedures for creating an admin file and using the -a option are discussed in the section "Setting Installation Defaults."

Troubleshooting Software Installation

The pkgadd command performs numerous checks as it installs a software package and, in a sense, performs troubleshooting for you. You will know a problem has occurred when:

- pkgadd displays a descriptive message. You should decide what actions to take based on the message. In most cases pkgadd will prompt you for instructions and wait.
- Installation results in a partially installed package. When this happens, you should attempt to reinstall the package.

Figure 14-2 describes some common installation errors and illustrates that in most cases you will need to either answer a prompt or attempt package reinstallation to correct the error.

Determining the Status of a Package

The status of a package will be either fully or partially installed. A package is considered fully installed, and is noted as such in the installation software database, when the system is notified that all actions needed to install the package have executed successfully. When installation completes or terminates and all actions have not taken place or some have been unsuccessful, then the package is considered partially installed and is noted as such in the database.

The status field in the display created by executing `pkginfo -l` *pkgid* shows whether the package was fully or partially installed.

Reinstalling a Package

Any time package installation results in a partially installed package, you should attempt to reinstall the package. If you know the reason it was only partially installed, correct the problem first and then reinstall.

You should be aware that reinstallation does not mean that every file associated with a package will be recopied or all actions reexecuted. When a package is in a partially installed state, `pkgadd` calculates at what point installation should resume and begins there. So, if you are installing a package with seven floppy disks, and a problem occurred when installing the last disk, you may not be required to repeat installation of all of the first six.

To reinstall a partially installed package, execute `pkgadd` as you did originally. For example, if you are installing a package from floppy disks, insert the first disk and execute `pkgadd`. You will receive instructions on how to proceed, including which disk needs to be inserted next. Because of the calculations made by `pkgadd`, the disk asked for may not be the one you expect. Be certain that you insert the disk that `pkgadd` has requested.

NOTE In some cases, you will have to remove a partially installed package before reinstalling it. If so, you will be notified when you execute `pkgadd`.

Figure 14-2: Common Installation Errors

Problem:	Accidental or purposeful termination of installation
Cause:	Break key used, program terminated, system difficulties
Fix:	Results in a partially installed package. Reinstall.

Problem:	Ran out of space on system during installation
Cause:	Inadequate space

Figure 14-2: Common Installation Errors (continued)

Fix:	Results in a partially installed package. Reinstall when space is available.
Problem:	Read error occurs during installation
Cause:	Corrupt media, hardware problems
Fix:	Attend to media or hardware problems. Attempt reinstallation.
Problem:	Content verification error
Cause:	Temporarily out of space, bad media, file changed by editing before check made
Fix:	Attempt reinstallation.
Problem:	Incorrect disk
Cause:	Installing disks out of sequence or using a disk not part of the package being installed
Fix:	Will receive message requesting that the correct disks be inserted. Follow the instructions supplied by `pkgadd`.

Installing Software from a Remote Machine: an Example with RFS

There are a number of software packages that can be installed from a remote machine. The following example provides instructions for doing so using the Remote File Sharing (RFS) utilities. For full details concerning RFS, or other remote software packages, refer to the guide that came with the software.

The example shows the steps the machineA administrator would follow to install an application that has been distributed on a cartridge tape onto machineA, which has no cartridge tape device. MachineB, connected to machineA via RFS, does have a cartridge tape device. In the following procedure, machineB shares its cartridge tape device with machineA so that machineA can install the application from the remote device.

Step 1: The administrator of machineB creates a directory called `sharedev` in the directory `/dev`, so that the entire `/dev` directory doesn't have to be shared:

```
mkdir /dev/sharedev
```

Step 2: The administrator of machineB then makes a link from the cartridge tape device (*ctape1*) to the shared directory device:

```
ln /dev/mt/ctape1 /dev/sharedev
```

Step 3: The administrator of machineB advertises the directory of shared devices (`sharedev`) to machineA; `machbtape` is used as the "resource identifier" for `sharedev`:

```
adv -d "B's cartridge tape" machbtape \
/dev/sharedev machineA
```

(This command should be entered on one line. It is shown here on two lines for readability.)

Step 4: The administrator of machineA can now create a directory called `machdev` in `/dev` for machineB's tape device, and then mount it:

```
mkdir /dev/machdev
mount -f rfs machbtape /dev/machdev
```

Step 5: Now the administrator can install the package *applpkg* from the cartridge tape inserted in the cartridge tape drive of machineB:

```
pkgadd -d /dev/machbdev/ctape1 applpkg
```

Checking Installation Accuracy

You may want to check the integrity of a package after it has been installed on your system. To do this, you use pkgchk. This command determines if an object has been modified by software or other actions since its installation. The command line for checking a package named pkgA would look like this:

```
pkgchk pkgA
```

You can name more than one package on the command line by separating the package names with spaces.

You can check specific pathnames, instead of all components of a package, by using the -p option to name the paths you want to check. If pkgchk is not given any name at all, meaning no package names or pathnames, then it checks the entire contents of a machine. You can name more than one pathname as long as the names are separated by commas (with no white space).

NOTE Packages must be identified by their package identifier. All instances of a package can be requested by adding .* to the package abbreviation. The PKGINST field on the pkginfo display shows the package identifier.

Defining the Type of Accuracy Check

pkgchk performs two kinds of checks. It checks file attributes (the permissions and ownership of a file and major/minor numbers for block or character special devices) and the file contents (the size, checksum, and modification date of a file). By default, the command checks both the file attributes and the file contents. You can check only the file attributes by using the -a option or only the file contents by using the -c option.

Checking Against the pkgmap File

The pkgchk command compares the file attributes and contents of the installed package against the installation software database. The entries concerning a package may have been changed since the time of installation. For example, another package may have changed a package component. The database will reflect that change.

If you want to compare the current integrity of a package to its integrity at the time it was originally installed, use the -m and -e options to specify the original description files. For example, if pkgA is mounted or spooled in the /install directory, the following command would be used.

```
pkgchk -m /install/pkgA/pkgmap \
      -e /install/pkgA/pkginfo
```

(This command should be entered on one line. It is shown here on two lines for the sake of legibility.)

Correcting Differences While Checking Accuracy

To correct file attributes when discrepancies are found, invoke pkgchk with the -f option. For example

```
pkgchk -f mypkg
```

attempts to correct any differences between the package components and the installation database or, if the -m and -e options have been used, between the components and the original pkgmap and pkginfo files.

Showing Information About Installed Packages

You can display information about installed packages by using the `pkginfo` command. It has a number of options that allow you to customize both the format and the contents of the display.

You can request any number of package instances if each name is separated by white space. If `pkginfo` is invoked with no packages named, it displays information for all completely installed packages on your system.

NOTE Packages must be identified by their package identifier. All instances of a package can be requested by adding .* to the package abbreviation. The `PKGINST` field on the `pkginfo` display shows the package identifier.

The Default `pkginfo` Display

When `pkginfo` is executed without options, it displays the category, package instance, and package name of all packages that have been completely installed on your system. The display is organized by categories as shown in the following example. (Package categories are defined by the package developer.)

```
$pkginfo
system        int       Installation Utilities
system        backup    Backup/Restore Utilities
application   pkgA      Package A
application   pkgA.2    Package A
application   anpkg     Another Package
$
```

Customizing the Format of the pkginfo Display

You can get a pkginfo display in any of three formats: short, extracted, and long.

The short format is the default format shown previously. It shows only the category, package abbreviation, and full package name. It presents one line of information per package.

The extracted format shows the package abbreviation, package name, package architecture (if available), and package version (if available). Use the -x option to request the extracted format as shown in the next example.

```
$pkginfo -x pkgA anpkg
pkgA          Package A
              (3B2) Release 2, Version 3
anpkg         Another Package
              (3B2) Release 4
$
```

Using the -l option produces a display in the long format showing all of the available information about a package, as in the following example.

```
$pkginfo -l mypkg
  PKGINST:      pkgA.3
     NAME:      Package A
 CATEGORY:      application
     ARCH:      3B2
  VERSION:      Version 3
 INSTDATE:      Tue Apr 14 08:41:40 MDT 1988
  BASEDIR:      /opt/pkgA
   VSTOCK:      sdr9000
   STATUS:      completely installed
    FILES:      31 installed
                3 linked
                10 dirs
                13 executable
SERIALNUM:     201-790b
$
```

The fields on this display are defined in the "Parameter Descriptions for the pkginfo Display" section.

Customizing the Contents of the pkginfo Display

You can use the following `pkginfo` options to specify packages to be included in the display:

- `-c` Select packages based on their category membership. For example, executing `pkginfo -c application` displays all packages belonging to the application category.
- `-i` Request that only completely installed packages be included in the display. For example, `pkginfo -i -c application` displays all completely installed packages belonging to the application category.
- `-p` Request that only partially installed packages be included in the display.
- `-a` Request that all packages with a particular architecture be included in the list.
- `-v` Request that all packages with a particular version be included in the list.
- `-d` Request that all spooled packages on a particular device, or in a particular directory, be included in the list. For example, `pkginfo -d /opt/spooldir -x` shows information in the extracted format for all packages in the spool directory `/opt/spooldir`.

NOTE The `-p` and `-i` options cannot be used in conjunction with `-d`, since `-d` implies "all packages spooled on this device" (or in this directory).

Parameter Descriptions for the `pkginfo` Display

The following list, organized alphabetically, describes all of the package parameters that can potentially be displayed for each package. A parameter and its value are only displayed when a package has a value assigned for it.

`ARCH`	The architecture supported by this package.
`BASEDIR`	The directory in which the software package resides (shown if the package is relocatable).
`CATEGORY`	The software category, or categories, of which this package is a member (for example `system` or `application`). (All packages produced before UNIX System V Release 4 are in the category `preSVR4`.)
`INSTDATE`	Date the package was installed.
`DESC`	Text that describes the package.
`EMAIL`	The electronic mail address for user inquiries.
`FILES`	Statistics on the number of files installed, partially installed, shared with other packages, or employing setuid objects.
`HOTLINE`	Information on how to receive hotline help concerning this package.
`NAME`	The package name, generally text describing the package abbreviation.
`PKGINST`	The package instance (package abbreviation plus suffix). Use this name in place of *pkgid* in any of the procedures described in this book.
`PSTAMP`	The production stamp for this package.
`SERIALNUM`	The serial number for this package.
`STATUS`	The package status, which can be installed, partially installed, or prior to UNIX System V Release 4.

VENDOR	The name of the vendor who supplied the software package.
VERSION	The version of this package.
VSTOCK	The vendor-supplied stock number.

Showing the Value of a Parameter

When you only want to know the value of one parameter, not the values for all of the parameters applicable to a package, use pkgparam as described below.

```
pkgparam -v pkgid param
```

where *pkgid* is the package identifier and *param* is the name of the parameter you want displayed. *param* must match the parameter definition, in most cases this means it should be entered in all capital letters. You can name multiple parameters when executing pkgparam as long as each is separated by white space. The -v option requests the verbose format, which shows the parameter name and its value. The following example displays the value of the basedir parameter.

```
$pkgparam -v pkgA BASEDIR
BASEDIR=/opt/pkgAdir
$
```

Storing Packages Without Installing Them

Storing a package means that the components of the package are copied directly from an installation medium to a spool directory, but no installation actions, such as running installation scripts or updating the installation software database, are taken.

Spooling a Package

To store packages without installing them, use the pkgadd command with the -s option. For example,

```
pkgadd -d diskette1 -s /var/temp/spooldir mypkg
```

would copy the package named mypkg from diskette drive diskette1 into a spool directory named /var/temp/spooldir.

When you follow the -s option with the word spool, pkgadd copies the package into the default spool directory (/var/spool/pkg).

NOTE You must use a package identifier with a numerical suffix if multiple versions reside on the installation medium. When there is only one version of a package on a medium, the package identifier is the package abbreviation without a suffix.

The package identifier suffix defines the package instance only on that particular medium. A new suffix will be assigned when this package is installed on your system. To find out what instances are available on a medium, run

```
pkginfo -d device
```

Checking Accuracy of a Spooled Package

You can use this command to check the accuracy of a spooled package instead of an installed package by using the -d option and naming the directory into which the package was spooled (or the device onto which it was spooled). The pkgchk command will look in this directory (or on this device) and perform its check. For example,

```
pkgchk -d spooldir pkgA
```

looks in the spool directory spooldir and checks the accuracy of a package named pkgA.

NOTE The checks made for a spooled package are limited since not all information can be audited until a package is installed.

Showing Information About Spooled Packages

You can request that all spooled packages on a particular device, or in a particular directory, be included in the `pkginfo` list by using the `-d` option. For example,

```
pkginfo -d /opt/spooldir -x
```

will show information in the extracted format for all the packages in the spool directory `/opt/spooldir`.

Removing a Spooled Package

The `-s` option of the `pkgrm` command removes a package from the spool directory. For example,

```
pkgrm  -s /opt/spooldir pkgA
```

removes `pkgA` from the spool directory `/opt/spooldir`. Naming only the spool directory and no packages will remove all spooled packages from the named directory.

Removing Packages

The pkgrm command removes both fully and partially installed packages from the system. It attempts to return the system to the same state it was in before the package was installed.

To remove a software package named mypkg from your system, you would enter

```
pkgrm pkgA
```

You can remove multiple packages by separating package names on the command line with white space, as follows:

```
pkgrm pkgA.3 mypkgB mypkgC.2
```

NOTE Packages must be identified by their package identifier. All instances of a package can be requested by adding .* to the package abbreviation. The PKGINST field on the pkginfo display shows the package identifier.

To remove a package in non-interactive mode, use the -n option. If there is a need for your interaction in this mode, pkgrm exits and removal fails.

Quick Reference to Software Management

- Setting installation defaults:

  Installation defaults are defined in an admin file. This file can be created with any editor and should contain a list of parameter definitions in the format of *param=value.*

  You can create this file in any directory; however, `pkgadd` normally looks in the `/var/sadm/install/admin` directory for admin files. If you put your admin file in this directory, you only need to supply the file name after the `-a` option when executing `pkgadd`. If you create your file in a different directory, you must supply the full pathname after the `-a`.

  The eleven parameters that can be set in this file, along with their description and possible values, are shown below.

  - `basedir` (package relocation information)

    directory name or `$PKG` or `$PKGINST`

  - `mail` (who should receive mail after installation)

    space-separated list of user IDs, defined as null (no mail sent), or not defined (mail is sent to root)

  - `runlevel` (run level dependencies not met)

    options: `nocheck, quit`

  - `conflict` (name conflict because a file or directory with the same name already exists)

    options: `nocheck, quit, nochange`

  - `setuid` (check for setuid or setgid execution)

    options: `nocheck, quit, nochange`

  - `action` (check security impact of package scripts)

    options: `nocheck, quit`

  - `partial` (partially installed version exists)

    options: `nocheck, quit`

- □ `instance` (instance already exists)

  options: `quit, overwrite, unique`
- □ `idepend` (package dependencies not met at installation time)

  options: `nocheck, quit`
- □ `rdepend` (package dependencies not met at removal time)

  options: `nocheck, quit`
- □ `space` (disk space requirements not met)

  options: `nocheck, quit`

Any parameter, except mail, may be defined as `ask`, meaning if the situation occurs you should be asked what to do.

- Storing interactions with a package

  `pkgask -d` *device* `-r` *response pkgid*

  where *device* is the device on which the package is stored, *response* is the name of the file into which your answers to package interaction will be written and *pkgid* is the package identifier of the package whose administrator interaction you wish to run. The file created with this procedure should be used in the procedure to install a package in non-interactive mode.

- Installing a software package:

  `pkgadd -d` *device pkgid*

  where *device* is the full pathname of the device or directory on which the package is stored and *pkgid* is the package identifier of the package to be installed. *device* can also be the device alias.

- Installing a software package in non-interactive mode:

  `pkgadd -n -d` *device pkgid*

  where *device* is the name of the device on which the package is stored and *pkgid* is the package identifier of the package to be installed. The `-n` option requests the non-interactive mode.

- Installing a software package in non-interactive mode with a response file:

  `pkgadd -n -d` *device* `-r` *response pkgid*

  where *device* is the name of the device on which the package is stored, *pkgid* is the package identifier of the package to be installed, and *response* is the full pathname of the response file to be used during the installation process.

- Installing a software package and using an alternative admin file:

  `pkgadd -d` *device* `-a` *adminfile pkgid*

  where *device* is the name of the device on which the package is stored, *pkgid* is the package identifier of the package to be installed, and *adminfile* is the name of the alternative admin file to be used. pkgadd looks in the `/var/sadm/install/admin` directory for *adminfile* unless you supply a full pathname.

- Checking the installation accuracy of an installed package:

  `pkgchk` *pkgid*

  where *pkgid* is the package identifier of the package you wish to check. You can name more than one package ID if the names are separated by white space. (If no package name is supplied, the entire contents of the machine will be checked.)

- Checking the installation accuracy of a specific pathname:

  `pkgchk -p` *pathname*

  where only *pathname* will be checked. You can name more than one pathname if the names are separated by commas.

- Checking the installation accuracy of only the file contents or only the file attributes of a package:

  `pkgchk [-a|-c]` *pkgid*

  where -a will check only the file attributes of a package and -c will check only the file contents. *pkgid* is the package identifier of the package to be checked.

- Checking the installation accuracy of a file against the original description files instead of the installation software database:

  ```
  pkgchk -m /install/pkgid/pkgmap \
        -e /install/pkgid/pkginfo
  ```

  where `-m` and `-e` check the package instance, *pkgid,* against the original `pkgmap` and `pkginfo` files from the package's distribution medium that is mounted or spooled at `/install`. (*This command should be entered on one line; it is shown here on two lines for readability).*

- Correcting file attributes as they are checked:

  ```
  pkgchk -f pkgid
  ```

  where *pkgid* is the package identifier of the package to be checked. When the `-f` option is used, the command will attempt to correct any differences in the file attributes between the package and the installation software database

- Spooling a package:

  ```
  pkgadd -d device -s spooldir pkgid
  ```

  where *device* is the name of the device on which the package is stored, *spooldir* is the name of the directory into which the package will be spooled, and *pkgid* is the package identifier of the package to be installed.

- Checking the installation accuracy of a spooled package:

  ```
  pkgchk -d spoolarea pkgid
  ```

  where *spoolarea* is the name of the directory into which the package was spooled (or the device onto which it was spooled) and *pkgid* is the package identifier of the package. You can name more than one package if the names are separated by white space.

- Showing information about installed packages:

  ```
  pkginfo pkgid
  ```

  where *pkgid* is the package identifier of the package for which you are requesting information. You can name any number of instances separated by spaces. The display shows the package category, package instance, and package name for the each instance requested.

Executing `pkginfo` without naming a package instance displays information about all fully installed packages on your system.

- Showing information about spooled packages or packages on a particular device:

  ```
  pkginfo -d device
  ```

  where *device* is the name of a device, directory, or spool directory. Using the `-d` option with `pkginfo` displays information about all packages on that device or in that directory.

- Removing a spooled package:

  ```
  pkgrm -s spooldir [pkgid]
  ```

  where *spooldir* is the name of the spool directory and *pkgid* is the name of the package to be removed. If no package identifier is supplied, all packages spooled in the named directory will be removed.

- Customizing the format of a `pkginfo` display:

  ```
  pkginfo [-x|-l] pkgid
  ```

  where *pkgid* is the package identifier of the package, or packages, for which you are requesting information. The `-x` option requests the extracted display consisting of the package instance, package name, package architecture (if available), and package version (if available). The `-l` option requests the long display that consists of all available information about a package.

  Executing `pkginfo` without the `-x` or `-l` options causes the information to be displayed in the default format that shows the category, package instance, and full package name.

- Customizing the contents of a `pkginfo` display:

  ```
  pkginfo [-c category] [-i|-p] [-n] [-a
      arch] [-v version] pkgid
  ```

  where the options define the contents of the `pkginfo` display as follows:

  - `-c` *category* includes all packages in the category defined

- `-i` includes only fully installed packages
- `-p` includes only partially installed packages
- `-a` *arch* includes all packages with the specified architecture
- `-v` *version* includes all packages with the specified version

pkgid is the package identifier of the package(s) for which you are requesting information.

- Showing the value of only one parameter:

  `pkgparam -v` *pkgid param*

  where *pkgid* is the package identifier and *param* is the name of the parameter, or parameters, you want displayed. Separate multiple parameters by white space. `-v` shows the parameter value with a label (for example, `BASEDIR='/opt/pkgAdir'`). Without the `-v` option, only the parameter value is shown.

- Storing a package without installing it:

  `pkgadd -d` *device* `-s` *spooldir pkgid*

  where *device* is the name of the device on which the package is stored, *spooldir* is the name of the directory into which the package should be read, and *pkgid* is the package identifier of the package to be read.

- Removing a package:

  `pkgrm` *pkgid*

  where *pkgid* is the package identifier of the package to be removed.

- Removing a software package in non-interactive mode:

  `pkgrm -n` *pkgid*

  where *pkgid* is the package identifier of the package to be removed. The `-n` option requests the non-interactive mode.

- Removing a software package and using an alternative admin file:

  `pkgrm -a` *admin pkgid*

  where *pkgid* is the package identifier of the package to be removed and *admin* is the name of the alternative admin file. `pkgrm` looks in the `/var/sadm/install/admin` directory for the admin file unless you supply a full pathname.

15. STORAGE DEVICE MANAGEMENT

15 Storage Device Management

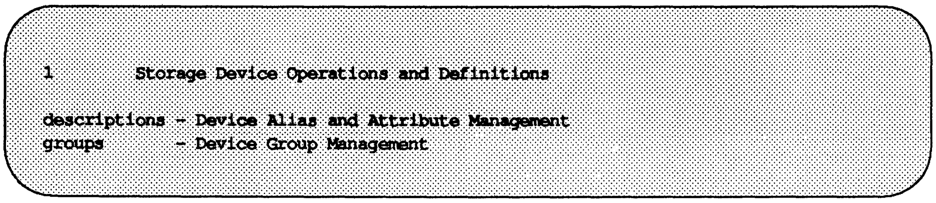
Introduction

This chapter tells you how to manage the disk and tape devices that are attached to your computer. You can do any of these functions by selecting the appropriate "task" from a series of menus provided for administration. To access the "system administration" menu for managing devices, type

```
sysadm storage_devices
```

The following menu will appear on your screen:

Figure 15-1: The Storage Device Management Menu



```
1          Storage Device Operations and Definitions

descriptions - Device Alias and Attribute Management
groups       - Device Group Management
```

In addition to the menus, there are many tasks you can perform by using shell level commands. The following table shows the shell commands that allow you to manage the storage devices on your system.

Task to Be Performed	Shell Command
Adding a device	`mknod(1M)`
Copying files to/from floppy diskette	`cpio(1)`
Copying files to/from cartridge tape	`ctccpio(1M)`
Copying files to/from SCSI tape	`cpio(1)`
Displaying contents of a cartridge tape	`ctcinfo(1M)`
Displaying contents of a disk	`mount(1M)`
	`ls(1)`
Duplicating disks	`dd(1M)`
	`cpio(1)`
	`dcopy(1M)`
	`volcopy(1M)`
Erasing disks and tapes	`format(1)`
	`rm(1)`
Formatting disks and tapes	`format(1M)`
	`fmtflop(1M)`
	`scsiformat(1M)`
Labeling file systems when copying to tape	`labelit(1M)`
Partitioning a hard disk	`mount(1M)`
	`fmthard(1M)`
Removing disks and tapes	`devnm(1M)`, `ls(1)`, `grep(1)`, `cp(1)`, `umount(1M)`, `rm(1)`
Verifying the usability of media	n/a
	`fsys(1M)`
	`fmtflop(1M)`
	`ctcfmt(1M)`
	`hdelogger(1M)`

Each task listed above is explained fully later in this chapter. Making duplicate copies of SCSI tapes is described in the "Copying Data on Storage Media" section of this chapter under the heading, "Special Cases." In addition, the *System Administrator's Reference Manual* and the *User's Reference Manual* provide manual pages for the shell commands.

The subjects of file systems and the information stored on disk devices are covered in the "File System Administration" chapter. The subject of bad block handling is covered in the "Diagnostics" chapter.

Overview of Managing Storage Devices

As a system administrator, you are responsible for controlling the resources of your system and who has access to these resources. One category of resources is the devices used for storing data, such as disks and tapes. There is a special file for each device in the `/dev` directory and there is a database that contains information about all devices on your system.

Each file has a special composition and, depending on the type of device it represents, resides either in the `/dev` directory or in a directory under `/dev`. How your system interacts with a device is defined by three attributes of the file in the `/dev` directory associated with that device: the name, the composition (that is, the major and minor device numbers), and the location of the file. Special files connect devices and device drivers. (The naming format, composition, and location of this identifying file is described in detail later in this chapter.)

Device Types

While there are several types of magnetic and optical storage devices available today, most 3B2 Computers commonly use a combination of hard disk, floppy diskette, cartridge tape, and 9-track reel-to-reel tape devices to store data. Terminals (ttys) are described in the "Service Access" chapter, printers in the "Print Services Administration" chapter, and communications in the "Network Services" chapter.

Hard Disk Devices

The UNIX operating system keeps all system software and user files on hard disks. Hard disks are available in a variety of sizes, providing a flexible range of storage capacity that allows you to add more devices as your user population and computing needs grow.

Floppy Diskette Devices

Floppy diskette drives are generally used to load software packages or user files onto the hard disk, to back up user files, and sometimes to back up file systems. Your 3B2 Computer comes with an integral floppy diskette drive (you can add others) that uses 5.25-inch, double sided (DS), double density (DD), 96 track per inch (TPI) floppy diskettes. When formatted, each diskette holds over 700 kilobytes (Kb) of data.

Tape Devices

Tape drives are used primarily for high-speed file system backups and file system or individual file restoration services. Your computer may be equipped with cartridge, 9-track, or other type of tape drives. Because of their high storage capacity, tapes provide an efficient way of storing data.

Small Computer System Interface Devices

Small Computer System Interface (SCSI) devices are a group of devices that adhere to the American National Standards Institute ANSI) standard for connecting intelligently-interfaced peripherals to computers. The SCSI bus is a daisy-chained arrangement originating at a SCSI adapter card that interconnects a number of SCSI controllers. Each controller interfaces the device to the bus and has a different SCSI address that is set on a switch located on the controller. This address determines the priority that the device is given, with the highest address having the highest priority.

SCSI storage devices include tape, hard disk, floppy diskette, and write-once-read-many (WORM) devices. SCSI tape devices are incompatible with tape devices that use similar media (such as cartridge tapes).

Because SCSI devices are daisy-chained together from a single adapter card, they do not require major hardware or software changes and are therefore easily adaptable to your computer. SCSI devices will work on most computers with UNIX System V Release 2.0.5, Release 3.0, and later software versions.

For all the information you will need to administer your SCSI device, you must refer to two sets of manuals: those for the computer and those for the SCSI device. Your 3B2 Computer has separate SCSI operation and installation manuals. For information about your SCSI device, refer to the operation and installation manuals for that device.

Device Identification Via Special Files

Before you can use any device with a computer running the UNIX operating system, the device must first be made known to the system. Devices delivered with your computer are automatically identified as the system is booted for the first time.

Directory listings for special files show two decimal numbers (called the "major" and "minor" device numbers) in the place where directory listings for regular files show the character counts. Figure 15-2 shows part of the output produced by running the `ls -l` command on a user's directory and on the directories under the `/dev` directory on a 3B2 Computer.

Figure 15-2: Directory Listings for a User's Directory and `/dev`

```
# ls -l
-rw-r-----    1 abc     other          1050 Apr 23 08:14 dm.ol
#
# ls -l /dev/dsk /dev/rdsk
/dev/dsk:
brw-------    2 root    sys     17,    0 Apr 15 10:59 c1d0s0
brw-------    2 root    sys     17,    1 Apr 12 13:51 c1d0s1
.
.
.
/dev/rdsk:
crw-------    2 root    sys     17,    0 Apr 15 10:58 c1d0s0
crw-------    2 root    sys     17,    1 Apr 12 13:51 c1d0s1
.
.
.
```

In Figure 15-2, file `dm.ol` is a regular file; all other files shown are device files. Four of the fields in this listing identify a file as a device: the first field, the fifth and sixth fields, and the last field.

- In the first field of this listing, the first character shows the type of file. Entries for regular files have a dash (–) in this position; entries for device files have the letters `b` (for block devices) or `c` (for character devices) in this position.

- For a regular file, the fifth and sixth fields show the character count (as one field); for a device file, these fields show the major and minor device numbers for the appropriate device. The major number specifies to the kernel which device driver is used for that device. The `drvinstall` command assigns major numbers to drivers that do not conflict with existing drivers. Major numbers for drivers are in the `/etc/master.d/README` file under the heading `External Major Numbers`. Major numbers serve as an index to the appropriate block or character switch table.

  The minor number specifies which device or subdevice is connected to the device driver. Minor numbers are passed to the device driver when a specific device driver function is called. The procedure for determining minor numbers for special files is device dependent.

  Block files can have the same major and minor numbers as character files. However, such file pairs either have different filenames or are in different directories (for example, `/dev/dsk/c0d0s0` and `/dev/rdsk/c0d0s0`).

- The last field shows (by its location in the `/dev` directory) how the system interacts with the device. For example, devices in the `/dev/dsk` directory are treated as block devices and devices in the `/dev/rdsk` directory are treated as character (or raw devices).

The following sections describe device types and partitions. For more information on the organization of device names and the file system, refer to the "Directories and Files" appendix of this book. For more information on the attributes of files and directories, refer to the "User and Group Management" chapter and the `ls`(1) command in the *User's Reference Manual.*

Block and Character Devices

All devices are either block type or character type; the classification of a device as one of these two types depends on how the device is accessed. When data are accessed in fixed-length blocks (that is, when the device does not permit access until a block of data has been accumulated), such a device is classified as a block device. Examples of block devices are disk drives and tape drives.

When data are accessed in chunks consisting of a specific number of characters (usually one), such a device is classified as a character device. File maintenance utilities may also use character devices. In the UNIX system, standard C language subroutines transfer data to these types of devices one character at a time. Examples of character devices are terminals and printers.

Most devices provide both character and block access; however, one type of access to a device is usually preferred. For example, a tape device has both types of access, but the preferred access type is block; character access to tape devices is possible but uses considerably more tape to store the same data, so block access is preferred. On the other hand, terminals prefer a character access type. Block access is possible, but the characters you type would not be echoed to the screen until you pressed a carriage return. The two special files for each device are explained next.

Summary

Devices are identified by special files in specific directories. The conventions used in positioning a device file depends on the type of computer and whether the device is internally or externally controlled. Standard file positions are used to identify the floppy diskette, hard disk, and cartridge tape devices. A distinction is made between character (raw) and block devices. Raw devices usually do not hold files or file systems and their names are positioned in the raw device directory (usually a `tty` assignment in the `/dev` directory that is linked to a file in the `/dev/rdsk` directory). Terminals, line printers, and tape drives are examples of raw devices. Block devices usually hold files and file systems and their names are positioned in the block device directory (usually `/dev/dsk` for disk devices). Floppy diskette and hard disk drives are examples of devices that are best accessed a block at a time.

The Device Alias

Every device has a device alias assigned to it. This alias is a unique name by which a device is known to the administrator. It is defined in the device database in `/etc/device.tab` (see "Managing Device Attributes" later in this chapter for more information). The device alias is mapped to the pathname.

Device Attributes

The device database, described in "Managing Device Attributes," should have an entry for every device. Each entry consists of a set of attributes and their value for that device. See "Creating a Device Entry" later in this chapter for a description of all possible attributes and what they mean.

The device entries should be created by the device installation script, if written for UNIX System V Release 4 or a later release. However, you will need to create entries for any device whose installation script does not so so.

Device Drivers

A device driver manages signals between a device and the operating system. If you want to design, install, and debug your own device drivers, there are several books available to help you do this. A few of them, written for advanced C language programmers, are:

- *Block and Character Interface (BCI) Driver Reference Manual*
- AT&T *Block and Character Interface (BCI) Driver Development Guide*
- AT&T *Portable Driver Interface (PDI) Reference Manual*
- AT&T *SCSI Driver Interface (SDI) Reference Manual*

Device Partitions

Devices can be partitioned to best support a system application. Certain partitions are used for specific functions when a device is a bootable device. The following list maps partitions to their use on a specific device type:

Partition 0	Used for root
Partition 1	Used for swap when the device is configured as the root device for the UNIX operating system
Partition 2	Used for the `/usr` file system

Partition 3	Used for the `/stand` file system
Partition 6	Specifies the entire device
Partition 7	Specifies the boot area of a device
Partition 8	Used for the `/var` file system
Partition 9	Used as the first hard disk area for user login directories (the default file system name is `/home`)
Partition 10	Used as the second hard disk area for user login directories (the default file system name is `/home2`)

Device partitions should fall on cylinder boundaries (that is, the same physical distance from the edge of the disk forming an imaginary "cylinder" through all disk segments) to obtain the best possible file system performance (the read/write heads do not have to move to find a partition; all partitions will be under the read/write heads at the same time). For a root device, the `boot` and `swap` partitions (partitions 7 and 1) are special in this regard. The number of blocks assigned to the `boot` and `swap` partitions are collectively chosen to cause the next partition values to fall on cylinder boundaries. (The next partitions are normally used as file systems.)

Floppy Diskette Partitions

Figure 15-3 defines the floppy diskette partitions in terms of use, starting sector, and total number of blocks for the various controller, device, and slice identifiers; they apply to both raw and block devices.

NOTE

The raw and block device partitions for the entire diskette (partition 6) are linked to `/dev/rSA/diskette1` and `/dev/SA/diskette1`, respectively. (Refer to the `link` command.) The use of these names when specifying the entire diskette is preferred over the use of the controller, device, and slice identifiers to avoid accidentally writing to a different device or partition.

The Volume Table Of Contents (VTOC) partitioning is not applicable to the diskette drive. (The VTOC is described in more detail in the "Formatting Floppy Diskette, Hard Disks, and Tapes" section of this chapter.)

Figure 15-3: Floppy Diskette Device Partitions

Disk Partition	Use	Sector Start	Total Sectors*
`c0d0s0`	root	378(21*18)	1044
`c0d0s1`	`usr`	612(34*18)	810
`c0d0s2`	`usr`	810(45*18)	612
`c0d0s3`	`usr`	1008(56*18)	414
`c0d0s4`	`usr`	1206(67*18)	216
`c0d0s5`	`usr`	18(1*18)	1404
`c0d0s6f1`	`entire disk`	0	1422
`c0d0s7`	boot	0	18

* Sectors are equivalent to 512-byte blocks

Hard Disk Partitions

There are up to sixteen different partitions (0 through 15) on a SCSI hard disk. Partitions 0 through 5 are reserved for system use and partition 6 defines the full user disk. Partition 7 is reserved for the boot blocks and the VTOC. You can assign partitions 8 through 15 for your users.

Tape Partitions

Figure 15-4 defines the partition, use, size, and number of blocks for the various controller, device, and slice identifiers for the 23-megabyte cartridge tape; they apply to both raw and block devices.

NOTE: VTOC partitioning is fixed by the tape formatting process.

Figure 15-4: Cartridge Tape Default Partitioning

Configuration	Partition	Use	Sector Start	Size*
Cipher,	c4d0s0	root	5272	8928
23-Megabyte	c4d0s1	swap	126	5146
Cartridge	c4d0s2	usr	14200	31341
Tape,	c4d0s3	usr	2	45539
(Gap=1),	c4d0s6	entire tape	0	45541
(Cyl=31)	c4d0s7	boot	0	126

* Size is in 512-byte blocks

NOTE

SCSI cartridge tapes do not have partitions.

Device Groups

You can define groups of devices to allow you to perform an action, or a set of actions, on several devices at the same time. For example, if you want to back up several devices on a regular basis, you can create a group for those devices. Then, whenever performing the backup operation, you can use the group name in place of a device name and every device in that group will be backed up. See "Managing Device Groups" for more information about device groups and how to create them.

Device Reservations

Devices can be reserved for exclusive use with the `devreserv` command. Reserving a device places it on a device reservation list and any new attempt to reserve that device will fail until the existing reservation is canceled. See "Managing Device Reservations" for details on how to create and manage device reservations.

Suggestions for Managing Storage Devices

Storage device management depends a lot on the size of your user population and how your resources are utilized. If there is a large user community with a lot of resources, managing your storage devices may take a lot of your time. The following are some suggestions for administering storage devices:

- When formatting floppy diskettes, format the entire box at once and label the box clearly so that you know that they are already formatted.
- When repartitioning a hard disk, make sure that you do a complete system backup before starting. This may be unnecessary if the hard disk contains a straight file system.
- If resources become depleted in one but not both of the `/tmp` and `/var/tmp` directories, ask users to distribute temporary files more evenly to both of these directories. (To do this, the `TMPDIR` environment variable may be defined in the user's profile as `TMPDIR=/tmp` or `TMPDIR=/var/tmp`.)
- When assigning a device for mounting the computer user file systems on a dual disk computer, such as the `/home`, `home2`, `home3`, etc., assign them to the second drive so that backups and restores are easier to do. However, allowing `/` and `/usr` to share the same disk can create performance problems.
- For a more secure computer, mount the `/usr` file system with read only permission.
- After copying valuable files from a floppy diskette to another medium, use the write-protect tab on the floppy diskette to prevent inadvertent erasure of the data.
- When copying large files, the value of `ulimit` may need to be changed to a number as large or larger than the character count of the largest file (`ulimit=`*number* where *number* is larger than the character count of the file as determined with `ls -l` *filename*).

Related administration may be in the form of software installation (see the "Software Management" chapter), backup and restore activities (see the "Backup Service" and "Restore Service" chapters), and disk failures (see the "Diagnostics" chapter).

Maintaining Devices and Media

With time, it may become necessary to change the number or type of storage devices connected to your computer. You will also need to perform some tasks, such as copying data and formatting diskettes, on a regular basis. This section describes the commands and steps associated with these tasks.

Adding a New Device

The need to add a new device to the system (that is, to define new special device files) occurs infrequently. If you add a device, the `autoboot` process defines the new files for you. However, if you cannot reboot the system and the new device uses an existing device driver, you can use the `/etc/mknod` command to define device files yourself.

NOTE If a new device driver must be installed along with the new device, you must reboot the system.

The syntax of `mknod` is:

```
mknod name b | c major minor

mknod name p
```

The or symbol (|) means that you must specify one or the other (b or c).

The options to `mknod` are:

name	The name of the special file.
b	A block device type.
c	A character device type.
major	The offset into the device driver (or controller) table in the kernel.
minor	The identifying number of this specific physical device.
p	A "named pipe" that functions as a first-in, first-out device.

To add a new block device:

1. `cd` to the directory where you want to install the new device (such as `/dev`)
2. Execute `/etc/mknod` *name* `b` *major minor*
3. Execute `chgrp root` *name*

To add a new character device:

1. `cd` to the directory where you want to install the new device (such as `/dev`)
2. Execute `/etc/mknod` *name* `c` *major minor*
3. Execute `chgrp root` *name*

Other device management techniques, such as attribute definitions, group information, and reservations are described in the "Machine Management" chapter.

Formatting Floppy Diskettes, Hard Disks, and Tapes

Before you can use a disk or tape for storing information, it must be formatted to impose an addressing scheme onto the media. For disks, formatting maps both sides of the disk into tracks and sectors that can be addressed by the disk controller. A portion of the disk is reserved for data having to do with the specific disk. The VTOC resides in that area and shows how the partitions on the disk are allocated. On a hard disk, another reserved area maps portions of the disk that may not be usable. Formatting a previously used disk, in addition to redefining the tracks, erases any data that may be there.

NOTE Unusable portions of the hard disk are called bad blocks. Bad block handling is discussed in the "Diagnostics" chapter.

Cartridge tapes are also divided into sectors and partitions but only one side of the tape is used for storing data. The action of passing the tape through the drive mechanism causes it to wear out eventually. So, an argument of the cartridge tape formatting command, `ctcfmt`, specifies the number of passes the tape can undergo before a warning message is issued. The default is 4000 passes.

NOTE SCSI tapes do not use pass numbering.

Diskettes are formatted by the UNIX system `fmtflop` command or by running `sysadm format`. You should format an entire box of diskettes when you first open the box to avoid the problem of keeping track of which ones are or are not formatted. If you adhere to this practice, you can assume that when you see an open box, all the diskettes in it have been formatted.

Cartridge tapes are formatted by the UNIX system `ctcfmt` command or by running `sysadm format`. When you have a cartridge tape device on your system, `sysadm format` asks you to choose which medium you want to format.

NOTE 9-track and SCSI tapes do not require formatting.

Hard disks are shipped from the factory already formatted.

NOTE The `fmthard` command does not format hard disks. This command is used to partition hard disk devices (other than the root device) and to install a VTOC.

During formatting, the `-v` option of the `fmtflop` and `ctcfmt` commands verifies that the formatting is being done without error. With `ctcfmt`, this option also builds a defect map.

For formatted diskettes, non-destructive data integrity can be verified with the `dd` command. For example, to find the number of whole and partial data blocks (input and output) on diskette1, run the `dd` command to copy data from `/dev/rSA/diskette1` to `/dev/null` (on completion, this command reports the number of whole and partial data blocks copied in and out), as follows:

```
# dd if=/dev/rSA/diskettel of=/dev/null
1422+0 records in
1422+0 records out
#
```

In this example, 1422 full and 0 partial records were copied in and out. The data block numbers you obtain depend on the size of the data blocks used on the disk. If the dd command stops in the middle of a copy, it means that it has found a bad block.

Displaying Information

NOTE The following commands may be used only on those 3B2 Computers equipped with cartridge tape drives.

To display information about a cartridge tape:

ctcinfo [*options*] *rawdevice*

where *options* are:

-a	Total number of bytes on a tape
-b	Total number of bytes per sector
-B	Total number of blocks on a tape
-c	Number of cylinders
-d	Device type
-m	Maximum tape pass count
-r	Reset tape pass count (used when tape heads are cleaned)
-s	Number of sectors per track
-t	Tape pass count
-u	Tape drive usage count
-v	Volume table of contents
-x	Number of tracks per cylinder

and *rawdevice* is the path to the raw device file of the cartridge tape device (such as /dev/rSA/ctapel)

To display device configuration information:

```
        dsconfig
```

To display volume table of contents information:

`prtvtoc` *device*

where *device* is the path to the raw device file (such as `/dev/rdsk/c2d0s6`).

To display device specific information:

`devinfo [-i | -p]` *special*

where `-i` displays the device name, drive ID number, number of device bytes per block, software version, number of device blocks per cylinder, and the number of device partitions with a block size greater than zero. The `-p` option displays the device name, partition start block number, device major and minor numbers, number of blocks allocated to the partition, and the partition flag and tag. The argument *special* is the path to the raw device (such as `/dev/rSA/disk1`).

To display major device numbers:

`getmajor` *name* | *ID_code*

where *name* represents a device found in the Equipped Device Table (such as SBD) and *ID_code* is the device number between 0x0 and 0xffff (hexadecimal).

To display a device name for a mounted file system:

`devnm` *filesystem*

where *filesystem* is the name of a mounted file system (such as `/usr`).

To display the number of free blocks and virtual nodes:

```
        df
```

You may also use `df` to display information about the generic superblock for mounted or unmounted file systems, directories or unmounted resources. (See `df`(1M) in the *System Administrator's Reference Manual.)*

Copying Data on Storage Media

There are two approaches to copying data placed on storage media. The first is to duplicate the entire medium onto another medium; the second is to copy specific files from one medium to another. If the file systems on both media are already mounted, you can use the cp command to perform either operation. If the file systems on both media are not mounted, you can use the `cpio`, `ctccpio`, or `dd` commands. Depending on the type of operation being performed, some commands work more efficiently and quickly than others. The preferred methods are described below.

To copy entire file systems quickly from hard disk to tape, you can use the `volcopy` command. This command is normally not used for small day-to-day copy operations.

Examples of each of these commands appear under "Quick Reference to Device Management" later in this chapter. Examples of the use of the `volcopy` command appear in the "Backup Service" chapter.

Copying Files from Hard Disk to Hard Disk

The cp command is commonly used when both the source and the destination file systems are already mounted. (See `mount`(1M) in the *System Administrator's Reference Manual.)* This command is generally used for copying small files quickly from one location to another.

Copying Files from Hard Disk to Floppy Diskette/SCSI Tape (and Vice-Versa)

When copying a large number of files to floppy diskette or to SCSI tape, it is often more efficient to use the `cpio` command because this command copies not only files, but also directories and subdirectories. (See `cpio`(1) in the *System Administrator's Reference Manual*).

Copying Files from Hard Disk to Cartridge Tape (and Vice-Versa)

When copying a large number of files to cartridge tape, it is often more efficient to use the `ctccpio` command because this command copies not only files, but also directories and subdirectories. (See `ctccpio`(1M) in the *System Administrator's Reference Manual*).

NOTE This command may be used only on those 3B2 Computers equipped with cartridge tape drives.

Copying Files from Floppy Diskette to Floppy Diskette

You can copy the contents of a source diskette to a destination diskette with the dd command. You can choose either a "character" or "block" device, but the same device (character or block) must be used for the entire procedure. With a single diskette drive, the data on the source diskette is copied to a temporary file on the hard disk in the temporary file space area. The source diskette is then replaced and the temporary file copied to the destination diskette. With multiple diskette drives, the source data is copied directly to the destination disk and this temporary file is not used.

NOTE While you can use any file system for temporary file space, it is wise to use space in either `/tmp` or `/var/tmp` since files in these two directories are automatically deleted during the transition to system state 2 (multi-user mode). At least 1422 free blocks must be available for temporary use in whichever directory you use.

Using the `fsck` Command

When attempting to mount a file system, you may get an error message saying that the file system is corrupt. You can use the `fsck` command to check the integrity of any mounted file system and, if possible, to repair it. (See `fsck`(1M) in the *System Administrator's Reference Manual.)*

Special Cases

You can make a copy of an entire SCSI cartridge tape. However, this requires that you sequentially copy the records off the tape into files on the hard disk. To copy the records into a single file, the `ulimit` may need to be changed. To change the value of `ulimit`, type, at the shell prompt, `ulimit` *number* where *number* represents the desired number of 512-byte blocks. See `sh`(1) for details. You must also have adequate free space on the file system to perform the copy (use the `df` command for this).

Figure 15-5 is an example of a shell script to duplicate a SCSI cartridge tape. This example can only be executed by root since the value of `ulimit` has been changed. Note the following:

- This example uses the home2 file system as the destination for the copied records. For this example to work, the directory /home/tmp must exist. If the df command shows that you do not have enough space in the /home2 file system to perform the copy, change the definition of "node" to the name of a file system that has adequate room.

NOTE: A translation between the block size used on your computer and bytes is required to ensure adequate space.

- The SCSI tape device is named c1t6d0s0n (a no-rewind device). If your SCSI tape device has a different device name, change the definition of "tape" on line number 1 to the name of your SCSI tape device.
- If the largest record on the tape is larger than 20480 blocks (10 Mb), the program will exit. Change the value of ulimit to a number larger than this size on line numbers 11 and 17.

Figure 15-5: Sample Script for Duplicating a SCSI Cartridge Tape

```
tape="/dev/rmt/c1t6d0s0n"
node="/home2/tmp"
count=0
mesg="Usage: $0 [-i | -o]"
if [ $# -lt 1 ]
then echo $mesg
     exit
fi
case $1 in
     -i) LIMIT=`ulimit`
         ulimit 20480
         while `dd if=$tape of=$node/$count bs=16k 2>/dev/null`
         do count=`expr $count + 1`
         done
         ulimit $LIMIT;;
     -o) LIMIT=`ulimit`
         ulimit 20480
         stop=`ls $node | sort -n | tail -1`
         while [ $stop -ge $count ]
         do dd if=$node/$count of=$tape bs=16k 2>/dev/null
         count=`expr $count + 1`
         done
         ulimit $LIMIT;;
     -*) echo $mesg; exit;;
esac
/usr/lib/scsi/tapecntl -y -w $tape
```

Name this executable file `scsicp` and move it to a restricted bin.

To make a copy of a SCSI tape, type `scsicp -i`. Once all records have been copied to the hard disk and your prompt returns, remove the source tape and replace it with the destination tape. Then, type `scsicp -o`. When your prompt returns, the duplication process is complete.

To make another duplicate of this tape, remove the destination tape, replace it with another destination tape, and type `scsicp -o` again.

Erasing Storage Media

Usually, when you want to remove files to create additional disk space, running the `rm` command is appropriate. However, when you want to erase all data, the procedure depends on the medium you want to erase.

Erasing Floppy Diskettes

To erase the contents of an entire floppy diskette, it is best to reformat the diskette. To do so, run the `fmtflop` command as follows:

```
fmtflop special
```

where *special* is the path to the floppy diskette block device (such as `/dev/SA/diskette1`).

Erasing a Cartridge Tape

To erase the contents of an entire cartridge tape, it is best to reformat the tape. To do so, execute

```
ctcfmt -v -t special
```

where `-v` verifies the formatting process and *special* is the path to the cartridge tape block device (such as `/dev/rSA/ctape1`).

Hard Disk Partitioning

The partitions on your computer's hard disk are allocated in a standard arrangement. This arrangement varies according to the numbers and sizes of the hard disks connected to your computer.

A single hard disk is partitioned to accommodate the root (`/`), `/usr`, `/stand`, `/var`, `/home`, `/opt`, `/share`, and other locally named file systems; swap space; and a small partition for the boot program. In multiple-disk systems, `/usr` is put on the second disk while root and `/home` share the first.

The default partitions are fundamentally a compromise. After your system has been in operation for a few months, you may feel that a different arrangement would better serve the needs of your users.

Planning to Change Hard Disk Partitions

The basic question to ask yourself when deciding to partition your hard disks is whether you should have a larger number of smaller file systems or stay with the default partitions. However, other questions also influence this decision. Some of these other questions are:

- What group IDs are defined? Are the right number of groups defined? Are users appropriately assigned to these groups? (See the "User and Group Management" chapter for more information on group IDs.)
- What type of processing is done by these groups? Does their work require temporary data storage? Is there a big difference between the type of processing done by one group and that done by other groups?
- Has software that affects the current plan for space requirements been added to the system? Will such software be added in the future?

The `sadp` command provides performance information about your existing file system arrangement and is described in the "Performance Management" chapter.

Hard disk partitioning can be done with a full system restore, as described in the "Backup Service" chapter. The `fmthard` command can be used to redefine partitions on disks other than the root disk.

If you find that your users need a large region of temporary space, you may want to create a separate partition for `/var/tmp`. If you do this, position the partition at the beginning of a disk different from your `root` and `/usr` partitions, if possible, to help balance your disk loads.

Changing Partitions to Increase Swap Space

If you frequently get console messages warning of insufficient memory, it may mean that either the amount of main memory or the swap area configuration is insufficient to support user demands. Before adding more main memory, you can first try to expand the swap area. There are three things you should do before expanding the swap area:

1. Identify the sizes of your present partitions. (Run `sysadm storage_devices` to obtain this information via a menu.)

2. Decide what the new partition sizes should be. (The disk is already fully allocated. Increasing the size of the swap partition means that you must reduce the size of another partition.)
3. Perform a complete system backup. (The process of changing partitions may erase the entire disk. Refer to the "Backup Service" chapter for information on backups.)

You are now ready to reload the operating system. See your computer installation manual for a description of how to perform this operation.

Removing Storage Devices

WARNING

Component failure may occur if you disconnect any storage device from your computer while power is on.

Performing this procedure will destroy the data (that is, the mounted file systems) on the device you want to remove. Make sure that you have backed up the contents of the disk before removing it. Refer to the "Backup Service" chapter for complete instructions on how to do this.

Do not remove the disk on which the root file system is mounted.

There are occasions when you will need to bring a device out of service, such as when a hard disk has a defect. Critical devices, such as those on which essential file systems are mounted, can never be removed from service. However, you may remove non-critical devices from service from the command line with the following procedure or you may select `remove` from the `storage_devices` menu (included with the `sysadm` command). Then, when the system is scheduled for powerdown, you may service the defective non-critical device.

NOTE

You should know the path to the block or character device that you want to remove before starting this procedure.

To remove a non-critical device from service:

1. Warn users that the device is to be taken out of service with the `/etc/wall` command. Make the announcement as specific as possible: explain which file systems will not be accessible or which services will not be available. Whenever possible, allow enough time for users to complete business on a device before removing it from service.

2. Execute `/etc/devnm /` to determine the device on which the root file system is mounted. Your screen should look something like the following:

```
# /etc/devnm /
/dev/dsk/c1d0s0 /
#
```

   The number at the end of the pathname reported (`c1d0s0` in this example) is the number of the partition on which the root file system is mounted.

3. To determine the major and minor device numbers on which the root file system is mounted, execute `ls -l` *special*. The value of *special* should be the partition number you received in Step 2 with the following change: the last digit of that number should be `6`. The new number is that of the partition used to access the entire disk.

   In this example, the partition number reported by `devnm` is `c1d0s0` so the value of *special* should be `c1d0s6`, as shown below:

```
# ls -l /dev/dsk/c1d0s6
brw-------   3 root    sys     17,     6 Feb 23  1988 /dev/dsk/c1d0s6
#
```

   The output shows that the major device number is `17` and the minor device number is `6`.

4. Execute `/etc/devnm /usr` to determine the device on which the user file system is mounted. Your screen should look something like the following:

```
# /etc/devnm /usr
/dev/dsk/c1d0s2 /usr
#
```

5. To determine the major and minor device numbers on which the user file system is mounted, execute `ls -l` *special*. The value of special should be the partition number you received in Step 2 with the following change: the last digit of that number should be `6`. The new number is that of the partition used to access the entire disk.

   In this example, the partition number reported by `devnm` is `c1d0s2` so the value of *special* should be `c1d0s6`, as shown below:

```
# ls -l /dev/dsk/c1d0s6
brw-------   2 root    sys     17,    6 Feb 23  1988 /dev/dsk/c1d1s6
#
```

   The output shows that the major device number is `17` and the minor device number is `6`.

6. Execute `ls -l` *device_path*

   where *device_path* is the path to the character or block device that you want to remove. Record the major and minor device numbers for this device.

   For example, if the device that you want to remove is the second internal hard disk drive in a 3B2/400, the path to this block device is `/dev/SA/disk2`. Your screen should look something like the following:

```
# ls -l /dev/SA/disk2
brw-------   2 root     sys       17,    22 Feb 23  1988 /dev/SA/disk2
#
```

The output shows that the major device number is 17 and the minor device number is 22.

7. Verify that the major and minor numbers of the device that you want to remove are not the same as those of the root or user file systems (see the output of the commands run in Steps 5 and 6 above).

NOTE If both the major and minor device numbers of this device match the major and minor device numbers of the devices on which the root and usr file systems are mounted, the device may not be removed.

8. Determine the target controller, drive, and hardware slot numbers of the disk by searching the /dev/dsk directory for the device name with the same major and minor device numbers (these numbers are coded into the device name in this directory). Use the grep command with the ls command as shown next.

```
# ls -l /dev/dsk|grep "17,     22"
brw-------   6 root     sys       17,    22 Dec 30 15:26 c1d1s6
#
```

Here, the target controller number is 1, the drive number is 1, and the hardware slot number is 6.

9. List the current directories mounted on this device with the grep command, as shown next.

```
# grep c1d1s6 /etc/vfstab
/dev/rdsk/c1d1s6 /dev/rdsk/c1d1s6 /home4   /home2   s5     -       yes     -
#
```

NOTE Keep a written record of these directories; they must be restored once the device is returned to service. (Refer to the "Restore Service" chapter for complete instructions on how to restore these directories from a backup tape.)

10. Save the current file system table by copying it, as shown next.

```
# cp /etc/vfstab /etc/Ovfstab
```

11. Unmount the directories from the device with the `/etc/umount` command, as shown next.

```
# /etc/umount /home4
#
```

12. Edit the `/etc/vfstab` file, removing unwanted references to the device. Remove the name of this device from the `/dev/dsk`, `/dev/rdsk`, `/dev/SA`, and `/dev/rSA` directories by running the `rm` command, as shown next.

```
# rm /dev/dsk/c1d1s6 /dev/rdsk/c1d1s6
# rm /dev/SA/c1d1s6 /dev/rSA/c1d1s6
#
```

13. If the device contains one or more file systems, remove all entries associated with this device from the `/etc/vfstab` file.

14. Remove the device from the device database by finding its alias and then removing the device as shown below. *device_path* is the pathname to a device.

```
# devattr -v device_path alias
alias='device_alias'
# putdev -d device_alias
#
```

15. Find out what groups the device belongs to and then remove the device name from the group membership list or lists in the device group database as shown next. *alias* is the alias name returned in the previous step.

```
# getdgrp alias=alias
device_group1
device_group2
# putdgrp -d device_group1 alias
# putdgrp -d device_group2 alias
#
```

Managing Device Attributes

The system stores information about devices in a database that can be accessed by applications that depend on device specific information. This section explains how to examine the information in this database, how to create new entries, and how to change or remove existing entries.

The Device Database

The device database resides in `/etc/device.tab`. It has one entry per device, consisting of a set of attributes that describe the device. This database is used by applications, such as the backup and restore services, which depend on device-specific information.

If you need to add, change or delete an entry, you can do so by using the `putdev` command, as described in the procedures to follow.

NOTE You can have a device installed that does not have an entry in this database. However, applications that access the device database for information will not be able to use such a device.

Creating a Device Entry

Each entry in the device database is composed of a list of attributes for a particular device. Every device must have the `alias` attribute assigned (this attribute defines the alias name for a device). No other attributes are required. The set of assigned attributes can, and probably will, vary from device to device. Figure 15-6 shows recommended values for the attributes of different types of devices.

Figure 15-6: Recommended Default Attribute Values

Alias	Description	Medium	Type
`9track`*N*	9-track Tape Drive *N*	tape reel	9-track
`ctape`*N*	Cartridge Tape Drive *N*	cartridge	ctape
`dpart`*N*	Disk Partition *N*	n/a	dpart
`disk`*N*	Integral Disk Drive *N*	n/a	disk
`diskette`*N*	Floppy Diskette Drive *N*	diskette	diskette
`qtape`*N*	QIC Tape Drive *N*	cartridge	qtape

n/a = not applicable

The following list defines the standard device attributes that can be defined for a device in the device database. However, there are no restrictions on what can be defined as an attribute. If you have an attribute you want to define, simply name it when invoking the `putdev` command.

Attribute	Description
`alias`	The unique name by which a device is known. No two devices in the database may share the same alias name. The name is limited in length to 14 characters and should contain only alphanumeric characters and also the following special characters if they are escaped with a backslash: underscore (_), dollar sign ($), hyphen (–), and period (.).

Attribute	Description
`bdevice`	The pathname to the block special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the `cdevice` field, if any. (It is your responsibility to ensure that these major/minor numbers are unique in the database.)
`capacity`	The capacity of the device or of the typical volume, if removable.
`cdevice`	The pathname to the character special device node associated with the device, if any. The associated major/minor combination should be unique within the database and should match that associated with the `bdevice` field, if any. (It is your responsibility to ensure that these major/minor numbers are unique in the database.)
`cyl`	Used by the command specified in the `mkfscmd` attribute.
`desc`	A description of any instance of a volume associated with this device (such as floppy diskette).
`dpartlist`	The list of disk partitions associated with this device. Used only if `type=disk`. The list should contain device aliases, each of which must have `type=dpart`.
`dparttype`	The type of disk partition represented by this device. Used only if `type=dpart`. It should be either `fs` (for filesystem) or `dp` (for data partition).

Attribute	Description
`erasecmd`	The command string that, when executed, erases the device.
`fmtcmd`	The command string that, when executed, formats the device.
`fsname`	The filesystem name on the filesystem administered on this partition, as supplied to the `/usr/sbin/labelit` command. This attribute is specified only if `type=dpart` and `dparttype=fs`.
`gap`	Used by the command specified in the `mkfscmd` attribute.
`mkfscmd`	The command string that, when executed, places a file system on a previously formatted device.
`mountpt`	The default mount point to use for the device. Used only if the device is mountable. For disk partitions where `type=dpart` and `dparttype=fs`, this attribute should specify the location where the partition is normally mounted.
`nblocks`	The number of blocks in the filesystem administered on this partition. Used only if `type=dpart` and `dparttype=fs`.
`ninodes`	The number of inodes in the filesystem administered on this partition. Used only if `type=dpart` and `dparttype=fs`.
`norewind`	The name of the character special device node that allows access to the serial device without rewinding when the device is closed.

Attribute	Description
`pathname`	Defines the pathname to an inode describing the device (used for non-block or character device pathnames, such as directories).
`type`	A token that represents inherent qualities of the device. Standard types include: 9-track, ctape, disk, directory, diskette, dpart, and qtape. See Figure 15-6 for a list of recommended attributes for these device types.
`volname`	The volume name on the filesystem administered on this partition, as supplied to the `/usr/sbin/labelit` command. Used only if `type=dpart` and `dparttype=fs`.
`volume`	A text string used to describe any instance of a volume associated with this device. This attribute should not be defined for devices which are not removable. See Figure 15-6 for example volume descriptions.

Use the `putdev` command to create an entry for a device in the device database. Execute

```
putdev -a alias [attribute=value [...]]
```

where *alias* is the alias name of the device to be added to the database and *attribute=value* is a list of attribute values to be associated with the device.

If the list of attributes described with the `putdev` command does not provide enough information for a device definition, you can use new attributes to satisfy the need. Such an attribute would be created simply by adding a definition for it to the attribute list of a device entry. (See the procedure entitled "Modifying a Device Entry.")

The following shows the command line required to add a device with the alias of `diskette3` to the database:

```
putdev -a diskette3 desc="Floppy Diskette Drive 3" type=diskette
```

Listing Devices

Use the `getdev` command to generate a list of devices. Executed without any options, this command creates a list of all devices in the device database. For example

```
getdev
```

might generate a list such as:

```
# getdev
ctape1
disk1
disk2
diskette1
spool
#
```

You can customize lists by naming the devices to be included, by defining the criteria that a device must match before being included, or by supplying both device names and a list of criteria. Using `getdev` like this allows you to obtain answers to questions such as the following.

- For what devices is a format command defined?
- What devices besides `spool` are set up in the device database?

Specifying Devices on the `getdev` Command Line

Name devices on the `getdev` command line by executing

```
getdev [-e] device [device [...]]
```

where *device* is the name of the device or devices that you want included in the list. All devices named will be included, unless you use the -e option, which specifies that the devices named should be excluded from the list.

Naming Criteria on the `getdev` Command Line

To name criteria on the `getdev` command line, execute

```
getdev [-a] criteria [criteria [...]]
```

where *criteria* is specified with expressions. The four possible expression types are:

attribute=value	Selects all devices for which *attribute* is defined and is equal to *value.*
attribute!=value	Selects all devices for which *attribute* is defined and does not equal *value.*
*attribute:**	Selects all devices for which *attribute* is defined.
*attribute!:**	Selects all devices for which *attribute* is not defined.

You can define a list of criteria simply by supplying more than one expression, each separated by white space. Devices must satisfy at least one criteria in the list unless the -a option is used. In that case, only those devices matching all criteria will be included.

Examples of Customizing the List of Devices

In each of the following examples, a user question is posed, followed by a `getdev` command line that you would enter to obtain the answer to that question. The resulting output would be a list of devices.

- What devices besides `spool` are set up in the device database? Execute:

```
getdev -e spool
```

- What devices have the fmtcmd attribute defined? Execute:

  ```
  getdev fmtcmd:*
  ```

- What devices do not have the fmtcmd attribute defined? Execute:

  ```
  getdev fmtcmd!:*
  ```

- What devices have the attribute type defined as disk or have the attribute part defined? Execute:

  ```
  getdev type=disk part:*
  ```

- What devices have the attribute type defined as disk and have the attribute part defined? Execute:

  ```
  getdev -a type=disk part:*
  ```

  (Note that this example differs from the previous one by requiring that a device adhere to both criteria, not just one.)

- What devices in the named list (disk1, disk3, and disk5) have the attribute type defined as disk or have the attribute part defined? Execute:

  ```
  getdev type=disk part:* disk1 disk3 disk5
  ```

Listing Device Attributes

The devattr command displays the attribute values for a device. The display can be presented in two formats:

- The default format shows a list of attribute values, without a descriptive label for each attribute.
- The verbose format, requested with the -v option, displays the attribute as attribute=value.

To list device attributes, type

```
devattr [-v] device [attribute [...]]
```

where *device* is the pathname or alias of the device whose attributes should be displayed and *attribute* is the specific attribute whose value should be displayed.

If you do not name a specific attribute, all attributes associated with the device are shown in alphabetical order.

For example, executing

```
devattr -v diskette1
```

produces the display shown below.

```
alias='diskette1'
bdevice='/dev/diskette'
cdevice='/dev/rdiskette1'
copy='true'
desc='Floppy Drive'
erasecmd='true'
fmtcmd='/etc/fmtflop -v /dev/rdiskette'
mkfscmd='/etc/mkfs -F S51K /dev/diskette 1422:512'
mountpt='/install'
type='diskette'
volume='diskette'
```

To see only the value for the `mountpt` attribute, type

```
        devattr diskettel mountpt
```

This produces

```
        /install
```

From this example, you can see that the value for the `mountpt` attribute for the device `diskettel` is `/install`.

Modifying a Device Entry

The `putdev` command can be used to modify existing attribute values for a device or to add new attributes to a device entry. To do so, execute

`putdev -m` *device attribute=value* `[`*attribute=value* `[...]]`

where *device* is the pathname or alias of the device entry being modified, *attribute* is the name of the attribute being modified, and *value* is the value that should be assigned to the attribute.

If the specified attribute currently exists for this device in the device database, `putdev -m` modifies the value. If the attribute does not exist, it is added and given the value *value*. The alias attribute cannot be changed with `putdev -m`. This prevents an accidental modification or deletion of a device's alias from the database.

To delete an attribute definition from a device entry, use the `-d` synopsis of the `putdev` command as follows.

`putdev -d` *device attribute*

where *device* is the name of the device entry from which an attribute definition will be deleted and *attribute* is the name of the attribute. For example, executing

```
        putdev -d diskettel volume
```

removes the attribute `volume` from the device entry for `diskettel`.

To delete the value of an attribute but keep the attribute in the device entry, use the same format as above with the following exception — assign the attribute the value of null. For example, to remove the value of the `volume` attribute while retaining `volume` in the device entry, execute

```
putdev -m diskette1 volume=""
```

Removing a Device Entry

The `putdev` command can be used to delete a device entry from the device database. To do so, execute

putdev -d *device*

where *device* is the pathname or alias of the device being deleted from the device database.

Managing Device Groups

You can create device groups to allow you to perform an action, or a set of actions, on any number of devices by giving only the device group name. As an example, think about a multiple device computer center. If there were several different rooms, each with a number of devices, groups could be created consisting of the devices located in each room. Device operations could then easily be done on a room-by-room basis by use of the group name instead of a device name.

The device group database resides in `/etc/dgroup.tab`. It has one entry per device group, consisting of a membership list. You can read this database, but you should not edit it directly. For this, you must use the `putdgrp` command, as described in the procedures that follow.

Creating a Device Group

Use the `putdgrp` command to create a device group. Execute

```
putdgrp group_name alias    [alias [...]]
```

where *group_name* is the name of the group you are creating and *alias* is the device alias of the member, or members, of the group. The following example creates a group called `disk` with two members (`disk1` and `disk2`):

```
putdgrp disk disk1 disk2
```

Listing Device Groups

Use the `getdgrp` command to generate a list of groups that are defined in the device group database. Executed without options, this command creates a list of all device groups. For example, executing `getdgrp` might generate a list such as the one shown on the next page.

```
# getdgrp
ctape
disk
diskette
#
```

You can customize lists by naming the device groups to be included, by defining the criteria that the member of a group must match before being included, or by doing both. Using `getdgrp` like this allows you to obtain answers to questions such as:

- What device groups do I have access to besides `disk`?
- What groups have devices with the attributes `fmtcmd` defined?

Specifying Device Groups on the `getdgrp` Command Line

Name device groups on the `getdgrp` command line by executing

`getdgrp [-e]` *group_name* `[`*group_name* `[...]]`

where *group_name* is the name of the device group or groups that you want included in the list. All named groups will be included, unless you use the `-e` option, which specifies that the groups named should be excluded from the list.

Naming Criteria on the `getdgrp` Command Line

To name criteria on the `getdgrp` command line, execute

`getdgrp [-a]` *criteria* `[`*criteria* `[...]]`

where *criteria* is specified with expressions. The four possible expression types are:

attribute=value	Selects all device groups with at least one member whose attribute *attribute* is defined and is equal to *value.*
attribute!=value	Selects all device groups with at least one member whose attribute *attribute* is defined and does not equal *value.*

*attribute:**	Selects all device groups with at least one member whose attribute *attribute* is defined.
*attribute!:**	Selects all device groups with at least one member that does not have the attribute *attribute* defined.

You can define a list of criteria simply by supplying more than one expression, each separated by white space. To be included in the list, at least one member of a device group must satisfy at least one criteria, unless the -a option is used. Then, only those device groups with a member matching all criteria will be included.

Examples of Customizing the List of Device Groups

In each of the following examples, a user question is posed, followed by a `getdgrp` command line that you would enter to obtain the answer to that question. The resulting output would be a list of device groups.

- What device groups do I have access to besides `ctape`?

  ```
  getdgrp -e ctape
  ```

- What device groups have members with the `fmtcmd` attribute defined?

  ```
  getdgrp fmtcmd:*
  ```

- What device groups have members that do not have the `fmtcmd` attribute defined?

  ```
  getdgrp fmtcmd!:*
  ```

- What device groups have members with the attribute `type` defined as `disk` or have the attribute `part` defined?

  ```
  getdgrp type=disk part:*
  ```

- What device groups have members with the attribute `type` defined as `disk` and have the attribute `part` defined?

  ```
  getdgrp -a type=disk part:*
  ```

(Note that this example differs from the previous one by requiring that a group have a member adhering to both criteria, not just one.)

- What device groups in the named list (group1, group3, group5) have members with the attribute type defined as disk or have the attribute part defined?

```
    getdgrp type=disk part:* group1 group3 group5
```

Listing the Members of a Device Group

To display a list showing the names of devices that belong to a group, execute

`listdgrp` *group_name*

where *group_name* is the name of the group whose member list should be displayed. For example,

```
listdgrp disk
```

might produce a list such as the following:

```
# listdgrp disk disk1 disk2 #
```

This output shows that the group `disk` is composed of two members: `disk1` and `disk2`.

Modifying a Device Group

The `putdgrp` command can be used to change group definitions by adding or removing a device from the group definition. To do so, execute

`putdgrp [-d]` *group_name alias* [*alias* [...]]

where *group_name* is the name of the group definition to be modified. *alias* is the device alias of the device to be added to the group definition or, if the `-d` option is used, the name of the device to be deleted from the group definition.

For example,

```
putdgrp disk disk3
```

would add the device `disk3` to the group `disk` and

```
putdgrp -d disk disk3
```

would remove the device `disk3` from the group `disk`.

Removing a Device Group

To remove a device group definition from the database, execute

`putdgrp -d` *group_name*

where *group_name* is the name of the device group definition to be removed.

Managing Device Reservations

Device reservation is provided to help manage device use. Reserving a device places it on the device reservation list. This list contains the name of any device that has been reserved and the process ID that requested the reservation.

When a process requests a reservation, the device reservation list is checked. If the device does not already appear in the list, then it is available, and is added so that any future request to reserve that device will be denied. When a reservation is canceled, the device name is removed from the device reservation list and thus is available for a new reservation.

It is important to note that use of the device reservation system is not mandatory and that, when it is used, device reservation does not place any constraints on access to the device. It is assumed that when a reservation fails, the person or process attempting the reservation will not use the device. However, there is no mechanism to prevent it. Also, processes that do not request a device reservation can use a device that is reserved, since such a process would not have checked the reservation status to find out if it was reserved or not.

As administrator, you can reserve a device for exclusive use, release the reservation once you are finished with it, and check the status of a device.

NOTE Device reservation activities can be done from within application programs. Unpredictable behavior may result, however, when applications collide on device usage. An example occurs when one application uses device reservation and another does not. Here, both may attempt to access the same device, and thus collide.

Reserving a Device

To reserve a device for exclusive use, enter

```
devreserv key device
```

Key is a positive integer that will be associated with this reservation and must be used later to free this particular reservation. The key should be unique. A suggested convetion is to use the process ID of the calling process as the key (for example, `devreserv` *key device*). *device* may be either the alias or pathname of a device that should be reserved or a list of devices. If *device* is a list, then the first device in the list that is available will be reserved.

Freeing a Reserved Device

To free a device reservation, enter

```
devfree key [device [...]]
```

where *key* is the key to which the device has been reserved and *device* is the alias or pathname of the device(s) that should be freed from reservation.

`devfree` can be executed with only the *key* argument, in which case all devices reserved to that process ID will be released.

Checking Device Reservation Status

You can check device reservation status in either of two ways. You can list all devices that are currently reserved or you can list all devices that are currently reserved to a particular key.

To list all devices that are currently reserved, execute

```
devreserv
```

To list all devices that are currently reserved to a particular key, execute

```
devreserv key
```

Quick Reference to Device Management

Adding a new device to the system:

1. `mknod` *name* `b | c` *major minor*

   where *name* is the name of the special file, `b | c` is the device type of which `b` is a block device (such as a disk drive) and `c` is a character device (such as a terminal), *major* is the offset into the device driver (or controller) table in the kernel, and *minor* is the identifying number of this specific physical device.

2. `mknod` *name* `p`

   where *name* is the name of the special file and `p` is a "named pipe" that functions as a first-in, first-out device.

Duplicating a disk:

`dd if=`*in_file* `of=`*out_file*

where *in_file* is the path to the source file(s) (such as `/dev/rSA/diskette1`) and *out_file* is the path to the destination file(s) (such as `/dev/rSA/diskette2`). The source and destination files can be located on any medium. This command can also perform file format conversions during duplication (such as EBCDIC to ASCII).

It is recommended that the source be write-protected where possible to prevent inadvertent erasure.

Displaying the contents of a floppy diskette (character device):

`cpio -it <` *special*

where *special* is the path to the character device (such as `/dev/rSA/diskette1`). This version of the command shows you a table of contents of the `cpio` file on the floppy diskette in the device.

Displaying the contents of a floppy diskette (block device):

1. `mount -r -v` *directory special*

   where *directory* is the path of the directory (that is, the mount point) in which you want the contents of this device to be mounted (such as `/install`), and *special* is the path to the floppy diskette block device (such as `/dev/SA/diskette1`).

2. `ls -l` *directory*

   where *directory* is the path of the directory of Step 1 above (such as `/install`).

3. `umount -v` *special*

   where *special* is the path to the floppy diskette block device of Step 1 above (such as `/dev/SA/diskette1`).

Copying files from hard disk to a floppy diskette:

Method 1:

ls file_1 file_2...file_N | `cpio -oc >` *special*

where *file_1* is the path to the first file (such as `/usr/bin/awk`), *file_2* is the path to the second file (such as `/bin/ls`), and *file_3* is the path to the third file (such as `/tmp/file`), that you want to copy to a diskette, and *special* is the path to the floppy diskette block device (such as `/dev/SA/diskette1`).

Method 2:

1. `mkfs -F s5 /dev/rsave 1422`

2. `labelit` *fsname special*

   where *fsname* is the mounted name of the file system (such as `usr`).

3. `mkdir fsname`

4. `mount -v` */fsname special*

5. `cp` *path/files /fsname*

   where *path* is the path to the files on the hard disk that you want to copy (such as `/usr/bin`) and *files* is the name of the file that you want to copy to floppy diskette (such as `*`).

6. `umount -v` *special*

Method 3:

```
cat file | cpio -ovc > special
```

where *file* is the path to the file (such as `/tmp/file`) that contains a list of paths to files (one per line) that you want to copy to a diskette, such as:

```
1  /var/tmp/file1

2  /var/tmp/file2

3  /var/tmp/file3
```

and *special* is the path to the floppy diskette block device (such as `/dev/SA/diskette1`).

Copying the contents of a directory from hard disk to the cartridge tape:

```
ls dirname | ctccpio -ov -T device
```

where *dirname* is the path of the directory you want to copy to diskette (such as `/tmp`) and *device* is the path to the tape (character) device (such as `/dev/rSA/ctape1`).

Copying the contents of a directory from hard disk to the SCSI tape:

```
ls dirname | cpio -ov > device
```

where *dirname* is the path of the directory you want to copy to diskette (such as `/tmp`) and *device* is the path to the tape (character) device (such as `/dev/rSA/qtape1`).

Copying the contents of a directory from hard disk to the floppy disk:

1. `mount -v` *dirname special*

   where *dirname* is the path of the directory you want to copy to diskette (such as `/tmp`), and *special* is the path to the floppy disk (block) device (such as `/dev/SA/diskette1`).

2. `cp` *dirname path*

   where *path* is the path to the files that you want to copy to diskette (such as `/usr/bin/*`).

Copying files from floppy diskette to the hard disk:

1. `cd` *dirname*

   where *dirname* is the path to the directory into which you want to copy the files (such as `/var/tmp`)

2. `cpio -idumv <` *device*

   where *device* is the path to the floppy disk (block) device (such as `/dev/SA/diskette1`).

Copying files from a cartridge tape to a hard disk:

1. `cd` *dirname*

   where *dirname* is the path to the directory into which you want to copy the files (such as `/var/tmp`)

2. `ctccpio -imudv -T` *device*

   where *device* is the path to the tape (character) device (such as `/dev/rSA/ctape1`).

Copying files from a SCSI tape to a hard disk:

`cpio -idumv <` *device*

where *device* is the path to the tape (character) device (such as `/dev/rSA/qtape1`).

Formatting a floppy diskette:

`fmtflop` *special*

where *special* is the path to the floppy disk (character) device (such as `/dev/rSA/diskette1`).

Removing a non-critical device:

WARNING Removing a non-critical device while the system is still powered up can have potentially disastrous results and should be done only in an emergency. This procedure should be performed only by experienced administrators.

1. `/etc/wall`
2. `/etc/devnm /`

   This produces a partition name.
3. Replace the last character of the partition name with the number 6 and use this value as *special* in Step 4 below.
4. `ls -l` *special*

   where *special* is the file representing the partition with 6 as its last character. Record the major and minor device numbers for this device (this is the root device).
5. `/etc/devnm /usr`
6. Substitute whatever partition number produced in Step 5 above with the number 6 and use this value as *special* in Step 7 below.
7. `ls -l` *special*

   Record the major and minor device numbers for this device (`/usr` device).
8. `ls -l` *device_path*

   where *device_path* is the path to the character or block device that you want to remove. Record the major and minor device numbers for this device (removed device). You will use the filename of this device in Step 16 below.
9. Ensure that the major and minor numbers of the device that you want to remove are not the same as those of the root or `/usr` device.
10. `ls -l /dev/dsk|grep "`*major,*[ ]**minor*`"`

    where *major* is the major device number and *minor* is the minor device number of the device that you want to remove. The filename found at the end of this listing will be used in Step 15 below.

11. `grep` *device_name* `/etc/vfstab`

    where *device_name* is the name of the device that you want to remove (filename of *device_path* in Step 8 above). Record the directories that are mounted on this disk.

12. `cp /etc/vfstab /etc/Ovfstab`

13. `/etc/umount` *directory*

    where *directory* is the name of the directory mounted on the disk that you want to remove (results of Step 12 above).

14. Edit the `/etc/vfstab` file and remove the references to the device that you want to remove.

15. `rm /dev/dsk/`*filename* `/dev/rdsk/`*filename*

    where *filename* is the filename found in Step 10 above.

16. `rm /dev/SA/`*filename* `/dev/rSA/`*filename*

    where *filename* is the value used in Step 8 above.

17. `devattr -v` *device_path* `alias`

    where *device_path* is the full pathname of the device to be removed. This command returns the alias name for this device.

18. `putdev -d` *alias*

    where *alias* is the alias name found in Step 17 above.

19. `getdgrp alias=`*device*

    where *device* is either the pathname or the alias of the device. This command outputs the alias of the device, which you will use in the next step.

20. `putdgrp -d` *device_group device_path*

    where *device_group* is the name of the group from which the device will be removed and *device_path* is the alias name of the device found in Step 17 above.

Verifying the usability and repairing of a disk:

`mount -v` *directory special*

where *directory* is the directory mount point and *special* is the path to the disk (block) device (such as `/dev/SA/disk1`).

If you get an error message saying that the data on the diskette need checking, there is a problem with the integrity of the file system. Use `fsck -y -D` *file-system* where *file-system* is the path to the file system that you want to check (such as `/usr`).

Creating a device entry:

`putdev -a` *alias* `[`*attribute=value* `[...]]`

where *alias* is the alias name of the device to be added to the device database and *attribute=value* is a list of attribute values to be associated with the device. Possible attributes are defined in the ''Managing Device Attributes'' section of this chapter.

Listing devices:

`getdev [-ae] [`*criteria* `[...]] [`*device* `[...]]`

where *device* is the name of the device or devices that you want listed and *criteria* defines selection criteria for the list.

If you use `getdev` with no options or arguments, you get a list consisting of all devices in the device database.

criteria is specified with the four expression types defined in the ''Naming Criteria on the `getdev` Command Line'' section of this chapter. Devices must match at least one of the criteria given to be included in the list. However, use the `-a` option to request that devices match all of the given criteria before being included in the list.

All devices named will be included in the list, unless you use the `-e` option, which specifies that the devices named should be excluded from the list.

Listing device attributes:

`devattr [-v]` *device* `[`*attribute* `[...]]`

where *device* is the pathname or alias of the device whose attributes should be displayed and *attribute* is this specific attribute whose value should be displayed. Only the attribute value is shown. Use the `-v` option to display attribute values in `attribute='value'` format.

Modifying a device entry:

`putdev -m` *device attribute=value* `[`*attribute=value* `[...]]`

where *device* is the pathname or alias of the device entry being modified, *attribute* is the name of the attribute being modified, and *value* is the value that should be assigned to the attribute.

Removing a device entry:

`putdev -d` *device*

where *device* is the pathname or alias of the device being deleted from the device database.

Deleting an attribute from a device entry:

`putdev -d` *device* `[`*attribute* `[...]]`

where *attribute* is the attribute definition to be removed and *device* is the pathname or alias of the device whose entry is being modified.

Creating a device group:

`putdgrp` *group_name device* `[`*device* `[...]]`

where *dgroup* is the name of the group you are creating and *device* is the pathname or alias of the member, or members, of the group.

Listing device groups:

`getdgrp [-ae] [`*criteria* `[...]] [`*group_name* `[...]]`

where *group_name* is the name of the device group, or groups, you want included in the list generated by this command. *criteria* defines selection criteria for the list.

If you use `getdgrp` with no options or arguments, you get a list consisting of all device groups.

criteria is specified with the four expression types defined in the "Naming Criteria on the `getdgrp` Command Line" section of this chapter. Device groups must have at lease one member matching one of the given criteria to be included in the list. However, use the `-a` option to request that device groups must have at least one member matching all of the given criteria before being included in the list.

All device groups named will be included in the list, unless you use the `-e` option, which specifies that the named device groups should be excluded from the list.

Listing the members of a device group:

`listdgrp` *group_name*

where *group_name* is the name of a group for which a list of members should be displayed.

Modifying a device group:

`putdgrp [-d]` *group_name device* [*device* [...]]

where *group_name* is the name of the group definition to be modified. *device* is the pathname or alias of the device to be added to the group definition or, if the `-d` option is used, the name of the alias to be deleted from the group definition.

Removing a device group:

`putdgrp -d` *group_name*

where *group_name* is the name of the device group definition to be removed.

Reserving a device:

`devreserv` *pid device*

where *pid* is the process ID to which the device should be reserved and *device* is the pathname or alias of the device that should be reserved.

Freeing a reserved device:

```
devfree pid [device]
```

where *pid* is the process ID to which the device has been reserved and *device* is the pathname or alias of the device, or devices, that should be freed from reservation.

Checking device reservation status:

```
devreserv
```

lists all devices that are currently reserved.

16 System Setup

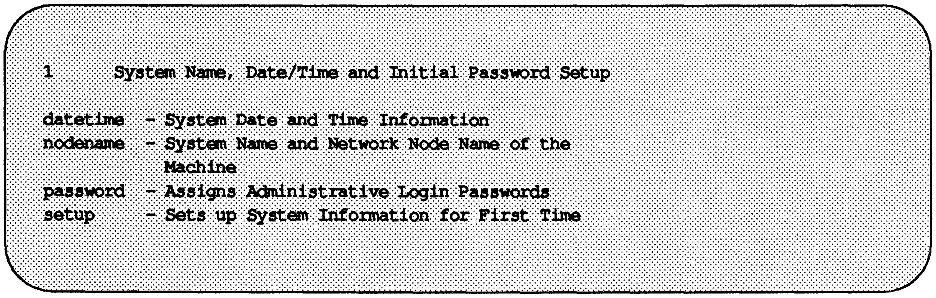
Introduction

This chapter tells you how to do some preliminary tasks that prepare your computer system for use. Specifically, it explains how to do the following:

- Set software options on your terminal
- Power up the computer
- Make a floppy key (to protect your system from unauthorized use)
- Set several system parameters, such as:
  - The name of your system
  - The date and time by which it runs
  - Create or change passwords for login names associated with specific administrative tasks

The first three tasks (setting terminal options, powering up the computer, and making a floppy key) must be done manually. The rest of these tasks, however, can be done manually or by using the system administration menus, a software interface provided by the UNIX system to help you with administrative work. To invoke the menu for system setup, type `sysadm system_setup`. The following menu will appear on your screen:

Figure 16-1: Main Menu for System Setup



```
1       System Name, Date/Time and Initial Password Setup

datetime  - System Date and Time Information
nodename  - System Name and Network Node Name of the
            Machine
password  - Assigns Administrative Login Passwords
setup     - Sets up System Information for First Time
```

If you prefer not to use the menus, you can do these system setup tasks by issuing the `setup` command and responding to a series of prompts. In addition, after you've established the initial setting for a parameter (such as the system name or an administrative password), if you decide that you want to change your setting, you may do so by running the appropriate shell command. The following table shows the shell commands that correspond to the tasks listed on the main system administration menu.

NOTE The `setup` command is listed for setting numerous parameters. However, it can be used for these settings only during the initial setup procedure. After the initial setup procedure, to change the parameter settings, you must use the alternate shell command for each parameter, appearing below the `setup` command in the following table.

Task to Be Performed	sysadm Task	Shell Command

Set up or display the system date and time	`datetime`	`date(1)`
		`setup(1)`

Set up the system name and nodename of a machine	`nodename`	`setuname(1M)`
		`setup(1)`

Display the system name and nodename of a machine	`nodename`	`uname(1M)`

Assign or change administrative passwords	`password`	`passwd(1)`
		`setup(1)`

Set up your machine for the first time	`setup`	`setup(1)`

Some of the menu items in the table above have their own underlying menus. This next table shows the menu items offered after the `datetime` menu item is chosen.

Task to Be Performed	sysadm Task	Shell Command
Display the system date and time	`display`	`date`(1)
Set up the system date and time	`set`	`date`(1)

This next table shows the menu items offered after the nodename menu item is chosen.

Task to Be Performed	sysadm Task	Shell Command
Display the system name and nodename of a machine	`display`	`uname`(1)
Set up the system name and nodename of a machine	`set`	`setuname`(1) `setup`(1)

Each command listed above is fully explained later in this chapter. In addition, the *System Administrator's Reference Manual* and the *User's Reference Manual* provide detailed descriptions of the shell commands.

Overview of System Setup

System setup begins where hardware installation ends. Once your computer and console terminal are installed, you are ready to set options on your terminal and administration parameters in the computer software so that your system functions smoothly and securely. This chapter provides instructions for the following steps in the system setup procedure:

- Selecting options for the console terminal
- Powering up the computer
- Creating a floppy key
- Setting system parameters
  - System date and time
  - User logins
  - System passwords
  - System name and node name

Because you may subsequently want to change the values of some parameters set during initial setup, this chapter also provides instructions for doing so. See "Changing System Parameters After Initial Setup" later in this chapter for details.

Additional Setup Tasks

After you've completed the setup procedure, you'll need to do a few other tasks before you're ready to let people start using the system. The following is a list of those tasks and the documentation for them.

- Install terminals for users - See the appropriate terminal manuals for instructions.
- Create file systems for users - See the "File System Administration" chapter in this book for instructions.
- Format floppy diskettes and cartridge tapes - See the "Storage Device Management" chapter in this book for instructions.

- Install local and remote printers - See the appropriate printer manual for instructions. You may want to connect a printer directly to the console terminal so you can print the information shown on your screen and save it for record keeping purposes.
- Install networking software - If you want to provide Remote File Sharing (RFS) on your system, install the Network Support Utilities (NSU) and the RFS Utilities and complete the setup procedure for network administration. See the "Network Services" chapter in this book for instructions.

System States

A "system state" is a set of conditions that allows the operating system to function in a particular way. For example, when the system is running in single-user state, only one user (an administrator with special privileges) can log in and issue commands. There are several system states; the UNIX system is always in one or another state when it is running.

Whenever the computer is turned on (including the first time), the system is in multi-user state by default. You may want to change this default to single-user state if you want to do start-up system testing or security checks every time you turn on the computer, before allowing conventional users to work on the system. To change the start-up system state default, use a text editor to modify the `initdefault` line in the `/sbin/inittab` file.

However, the first time you power up the machine (and on subsequent occasions if you don't change the default start-up state), you will need to change the system state to do some startup tasks, as well as other types of administrative work, that can be done only in single-user state. For instructions on changing system states, see the "Machine Management" chapter.

Setting Up the Console Terminal

Before powering up your computer, turn on the console terminal and select the following options for it:

- 9600 baud
- Full duplex
- Parity of "NONE" or space
- Check parity of "NO"
- 8-bit ASCII character set
- If there is an option for flow control, set the XON/XOFF software flow control (DC1/DC3).

For instructions on setting these options, refer to your terminal manual.

Powering Up the Computer

After your console terminal is set up correctly, you can power up your computer by performing the following steps:

1. Turn on the console terminal and wait for the cursor to appear.
2. After the cursor appears, turn on the computer according to the directions in your computer installation manual.

After a few minutes, your terminal should display start-up messages. The system startup is complete when the `Console Login:` prompt appears. If this prompt does not appear, try rebooting the system. If you still do not receive this prompt, call for service.

Remember, when you power up your system it will enter multi-user state by default. The boot up sequence is transparent to the administrator. For more information on system states, boot programs, and powering down your computer, see the "Machine Management" chapter.

Making a Floppy Key and Assigning a Firmware Password

Firmware is a set of programs and data stored in Non-Volatile Random Access Memory (NVRAM) that allows the administrator access to the machine at the firmware and hardware levels. Because of the need to restrict casual access to the firmware, the firmware is protected with a password.

The default firmware password is mcp. We recommend that you change the password, but not until after you've made a "floppy key." A floppy key is like an extra key to your house: if you later "lock yourself out" of your system by forgetting the firmware password you've assigned —or in the unlikely event that your firmware data become corrupted— you can use your floppy key to restore the default password and thus "get back in" to the system. For this reason, it's important to make a floppy key before changing the firmware password.

What exactly is a floppy key? A floppy key is a floppy disk that contains the serial number of your computer and certain information contained in the NVRAM. When used, the floppy key resets several system parameters to their original values. In addition to the firmware password, the floppy key resets the system name and the node name.

NOTE Because anyone who has access to the floppy key can access your operating system, it's important to store your floppy key in a safe place.

The rest of this section provides instructions for making a floppy key, changing the firmware password, and using the floppy key.

Making a Floppy Key

You can use any formatted disk to make a floppy key. For your convenience, a blank, formatted floppy disk labeled "Floppy Key" is delivered with the computer. If you use a floppy disk of your own, make sure it is write enabled. (If you are not familiar with using floppy disks, see your computer installation manual.)

To make a floppy key, follow these steps:

Step 1: Obtain a formatted floppy disk; if you do not have one that is formatted, use the `fmtflop(1M)` command to format a floppy disk.

Step 2: If you have not done so already, power up the system.

Step 3: When you see the `Console Login` prompt, type `root` and press RETURN.

Step 3: When you see the `Password:` prompt, enter the root password and press RETURN. The system displays the # prompt. (If there is no root password, the `Password:` prompt will not appear.)

Step 4: Bring the system to firmware mode through the `sysadm machine` menu, or using the `shutdown -i5` command. The system broadcasts a message to any users currently logged on instructing them to log off now. Then it stops all system services and displays the message `FIRMWARE MODE`.

Step 5: The following screen shows what is displayed in firmware mode and how you should respond to create a floppy key. The `newkey` command is used to start the process; you will need to type `go` once you insert the floppy disk into the drive.

```
FIRMWARE MODE

password

Enter name of program to execute [   ]: newkey

Creating a floppy key to enable clearing of saved NVRAM information

Insert a formatted floppy, then type 'go' (q to quit): go

Creation of floppy key complete


Enter name of program to execute [   ]:
```

Step 6: Remove the floppy disk from the disk drive, label it, and store it in a safe place.

Now that you have a floppy key, you can change your firmware password. We strongly recommend that you do this immediately. (See the instructions in the following section.) If you don't want to change the firmware password now, return the system to a normal operating state by following the procedure described in the "Machine Management" chapter under "Returning from Firmware."

Changing the Firmware Password

After you have made a floppy key, you can safely change the firmware password on your computer. (See the previous section for instructions on how to make a floppy key.) Before you do, read the guidelines about selecting a secure password in "Choosing a Password" in the "Security" chapter.

To change the firmware password, change the system to the firmware mode and follow these steps:

Step 1: At the firmware prompt (`Enter name of program to execute [] :`), type `passwd` and press RETURN. The system will respond: `enter old password:`

Step 3: Enter mcp (or the current password if you have previously changed it from the initial default). The system will respond: `enter new password:`

Step 4: Enter a new password. The system will respond: `confirmation:`.

Step 5: Enter the new password again. The system will respond: `Enter name of program to execute [ ]:`

Step 5: Return the system to a normal operating state by following the procedure in the "Returning from Firmware" section of the "Machine Management" chapter.

Using the Floppy Key

If you ever forget the firmware password, you can use your floppy key to restore the default firmware password `(mcp)`.

Step 1: Power down the computer.

Step 2: Insert the floppy key in the disk drive and turn on the computer. The NVRAM parameters will be reset and the firmware prompt (`Enter name of program to execute[]:`) will appear.

Step 3: Return the system to a normal operating state by following the procedure in the "Returning from Firmware" section of the "Machine Management" chapter.

The System Setup Procedure

To set your system parameters, type `sysadm setup`. You will be prompted to provide values for the following system parameters:

- Date, time, and time zone
- System name and node name
- Administrative passwords
- User logins

To change any of these parameters later, follow the appropriate instructions in "Changing System Parameters After Initial Setup" later in this chapter.

Testing the Results of the System Setup Procedure

1. After the setup procedure is finished, and the `Console Login:` prompt appears, log in using the conventional user ID that you assigned to yourself.
2. After the `$` prompt appears, test the system setup by entering the following commands, one at a time, and noting the responses:

`date`	Current date
`who`	Logins of all active users ID
`pwd`	Present working directory
`uname`	System name and node name

   Here's an example test of setup:

```
$ date
Wed Mar  1 12:17:00 EST 1989
$ who
user1  console  Mar  1 12:17
$ pwd
/home/user1
$ uname -s
UNIX System V
$ uname -n
mach1
```

3. After you have received correct responses, reboot your machine (according to the instructions in the "Machine Management" chapter) and wait for the `Console Login:` prompt. The reboot is necessary because some of the parameters set during setup will take effect only after a reboot.
4. If the `Console Login:` prompt does not appear, try rebooting again. If, after the second reboot, it still does not appear, call for service.

NOTE Never reboot your machine while you are in single-user state; first return to multi-user state and then reboot. If you accidentally reboot while in single-user state, you will have to change the speed of the terminal to 300 in order to reestablish communication with your computer.

Changing System Parameters After Initial Setup

The system setup tasks that you do by invoking the `setup` command, such as setting the system node name, are usually done only once, when you first set up your system. However other jobs, such as assigning administrative passwords, must be done intermittently, as necessary. The following situations are examples of ongoing administrative work that involves tasks described here as part of the setup procedure.

- As system passwords and administrative command passwords age, you must change them.
- If your system has been corrupted or down for any period of time, you must reset the system date and time.
- When users leave your system, you should change system passwords, administrative passwords, and any security passwords.
- If your system has been added to a network, and your existing system name and node name are already being used by another system, you should change your system name and node name.

This section describes how to add or change system parameters that you set during initial setup after you've completed the initial setup. The instructions in this section explain how to do the following:

- Change the system date and time
- Add user logins
- Change user passwords
- Change administrative passwords
- Change system passwords
- Change the system name and node name

Changing the System Date and Time

1. If your system is in multi-user state, shut it down to single-user state (see the "Machine Management" chapter for instructions).
2. Set the date and time for your system by selecting the `datetime` item from the `system_setup` menu or by issuing the `date` command as follows:

   ```
   date MMddhhmm[yy]
   ```

   where *MM* is the month, *dd* is the day, *hh* is the hour (in 24-hour clock format), *mm* is the minute, and *yy* is the last two digits of the year (with the current year being the default). For example, to set the date to October 6, 12:45 A.M., of the current year, enter the following:

   ```
   date 10060045
   ```

   Adjustments for daylight savings time will be made automatically once you have set the system date and time, regardless of whether you used the `date` command or the `datetime` menu item to do so.
3. To change the time zone for your computer, use the `datetime` menu item.

Adding User Logins

Before conventional users can log in, they must be added to the system. The `setup` command prompts you to add login names for the users on your system. If during the setup procedure, you forget to add a login name for a user or if you later want to add a new login, you can do so by running the `useradd` command. The simplest form of this command is

```
useradd login_name
```

You will also need to create a home directory and initial password for the new user. Details about the rest of the options for the `useradd` command are provided in the "User and Group Management" chapter. For more information on how to add user logins and passwords, and arrange for password aging, see the "Security" chapter.

If you prefer to use the system administration menus, you can add a user to your system by selecting the add item from the `users` menu. The add menu item will prompts you to create user ID entries.

Changing Users' Passwords

When a user forgets his or her password, you need to change it. To change a password, log in as `root` and type:

```
passwd login_name
```

where *login_name* is an administrative login, a system login, or a user's login.

You will be prompted for the old password (if there is one) and then for a new password. After you enter it, you will be prompted to re-enter the new password for verification. (Passwords are not displayed on the screen.)

If you are logged in as root, you may change the password for any type of ID. Users of any other login names can change only their own passwords.

For more information about managing users' passwords, refer to the "User and Group Management" chapter and the "Security" chapter.

Creating or Changing Administrative Passwords

Many administrative commands are used not only by the system administrator, but by conventional users, as well. However, giving users access to the `root` login so that they can use these commands is undesirable for security reasons. To allow access to administrative commands without allowing use of the `root` login, assign passwords for the commands themselves, just as you assign passwords for user logins. You can assign a password to a command by using the `passwd` command or through the `system_setup` menu. Once a command is assigned a password, any user attempting to execute it will be prompted for that password. Make sure that only a few users know the passwords.

During the system setup procedure (invoked by running the `setup` command), you are given the opportunity to assign passwords for various administrative commands. You may want to create or change passwords after you've finished setup, however. You may have forgotten to create a particular password. Or it may simply be time, as a result of password aging, for a routine change of passwords. You can create a new password or change an existing one by running the `passwd` command.

If you want to create more than a few passwords (or if you don't know which commands still require passwords), you may want to use the system administration menu called `system_setup` instead of using the `passwd` command. If you want to change a password, you don't have the option of using the menus; you must use the `passwd` command.

Instructions for creating and changing passwords with the `passwd` command and the `system_setup` menu are provided below.

Using `passwd` to Protect Administrative Commands

To create a new password for a login, or to change an existing one, run the `passwd` command as follows:

`passwd` *login_name*

If a password already exists, you will be prompted to enter it. Then you will be prompted for the new password and, after you enter it, you'll be prompted to re-enter it for verification. Keep in mind that passwords are not displayed on the screen.

Using `system_setup` to Protect Administrative Commands

If you don't know which administrative commands have passwords assigned to them, or if you want to create passwords for more than a few of them, use the system administration menu `system_setup`. Using this menu is more convenient than issuing the shell command for each administrative command that needs to be changed.

When you select `password` from the `system_setup` menu, you will be prompted to choose a password (if none exists) or leave the current password unchanged for each administrative login. You cannot use the menus to change an existing password. To change an existing password, you must the `passwd` command (see the instructions in the previous section).

Creating or Changing System Passwords

The UNIX system provides login names for eight important directories and commands that are usually used by administrators but that are sometimes needed by users, as well. By using one of these logins, a user can access the desired directory or command and issue the appropriate command. When the command has finished executing, the user will be returned to the shell, or to a `login:` prompt if the user logged in initially with an administrative login.

To avoid compromising security while making these resources available to users without special privileges, you can create passwords for these login names. Once a login is assigned a password, any user attempting to use it will be prompted for that password. Make sure that only a few users know the passwords.

During the system setup procedure (invoked by running the `setup` command), you are given the opportunity to create passwords for these special logins. You may want to create or change passwords after you've finished setup, however. You may have forgotten to create a particular password. Or it may simply be time, as a result of password aging, for a routine change of passwords. You can create a new password or change an existing one by running the `passwd` command.

If you want to create more than a few passwords (or if you don't know which commands still require passwords), you may want to use the system administration menu `system_setup` instead of using the `passwd` command. If you want to change a password, you don't have the option of using the menus; you must use the `passwd` command.

Instructions for creating and changing passwords with the `passwd` command and the `system_setup` menu are provided below.

Using `passwd` to Protect System Logins

To create or change a password for a system directory login or a system command login, type:

`passwd` *login_name*

where *login_name* is one of the following:

`root`	This login has no restrictions on it and it overrides all other logins, protections, and permissions. By entering the command `su root` at the system prompt, or typing `root` at the login prompt, a user can gain access to the entire operating system. The password for the `root` login should be very carefully protected.
`sys`	Owns many system files.
`bin`	Owns most of the commands in `/usr/bin`.
`adm`	Owns some administrative files in `/var/adm`.
`uucp`	Owns some administrative files in `/usr/lib/uucp`.
`nuucp`	Used by remote machines to log in to the system and start file transfers from `/var/spool/uucppublic`.
`daemon`	Login of the system daemon, which controls background processes.
`lp`	Login that owns the object and spooled data files in `/home/lp`.

Running the `passwd` command is convenient if you already know which logins need passwords, and you want to assign passwords to only a few of them. If you don't know which commands need passwords or if you want to create passwords for many of them, see "Using `system_setup` to Protect System Logins" below.

Using `system_setup` to Protect System Logins

If you don't know which system logins already have passwords assigned to them, or if you want to create passwords for more than a few of them, use the system administration menu `system_setup`. Using this menu is more convenient than issuing the shell command for each login requiring a password. For instructions, see "Using `system_setup` to Protect Administrative Commands" earlier in this section.

Changing the System Name and Node Name

If your computer is part of a network, you must define a system name and a node name for your computer. The node name must be unique within the network. These names are defined by assigning values to the system name and node name parameters. Default values for these parameters are delivered with the system and can be used as long as your system is not part of a network. If your system is going to join a network, however, you must change the node name.

Begin by reading the guidelines for selecting a system name and a node name in the "Network Services" chapter. Once you have selected names, you can assign them to the system name and node name parameters through either of two methods: by using the system administration menus or by issuing a command.

If you prefer to use the menus, select the `nodename` item from the `system_setup` menu. You will be prompted for a node name and a system name.

Otherwise, run the `setuname` command with the `-s` and `-n` options, as follows:

```
setuname -s system_name -n node_name
```

You can run this command from either multi-user state or single-user state but, in either case, you must be logged in as `root`.

System Information

To look up the existing system name and node name, use the uname(1) command with the options `-s` and `-n` as follows:

```
uname -s -n
```

The system will respond by displaying two names, as in this example:

```
UNIX localB
```

Here the system name is `UNIX`, and the node name is `localB`.

In addition to this information, you might want to look up other facts about your system. Which release of the `UNIX` system is it running? What kind of processor does it use? You can get answers to these types of questions by running various options to the uname command. The following is a list of the available options and a description of the information provided by each.

-s	System name
-n	Node name
-r	Operating system release
-v	Operating system version
-m	Computer (hardware) name
-p	Processor type
-a	All of the above information

The -a option provides a convenient way of getting most of this information at once. Simply by typing

```
uname -a
```

you can receive a one-line report such as the following example:

```
UNIX localB 3.0V2 3 3B2 M32100
```

Here `UNIX` is the system name, `localB` is the node name, and `3.0V2` is the operating system release. The `3` represents both the operating system version associated with this system, and the node ID for it. (In this example, the `3` identifies this system as the third system and node ID assigned to your machine). `3B2` is the name of your computer, and `M32100` is the processor type.

Quick Reference to System Setup

- Change passwords for users or administrative functions:

  `passwd` *login_name*

- `Display the system name and node name:`

  ```
  uname -a
  ```

- List the users currently logged on to the system:

  ```
  who -H
  ```

- Perform an interactive setup:

  ```
  setup
  ```

- Return to multi-user state:

  ```
  init 2
  ```

- Send a message to users logged on now:

  ```
  /usr/sbin/wall
  ```

- Set or change the system name:

  `setuname -s` *system_name*

  where *system_name* is an alphanumeric string that contains no more than eight characters.

- Set or change the node name:

  `setuname -n` *node_name*

  where *node_name* is an alphanumeric string that contains no more than eight letters.

- Set or reset the system clock:

  `date` *MMddhhmm*[*yy*]

  where *MM* is the month, *dd* is the day in the month, *hh* is the hour (24-hour system), *mm* is the minute, and *yy* is the last two digits of the year with the current year as the default. (The *yy* argument is optional.) To set your local time zone, you must use the `datetime` menu item. The `date` command adjusts the system date and time as necessary to accommodate daylight savings time.

- Set the console terminal options:
  - 9600 baud
  - Full duplex
  - Parity of "NONE" or space
  - Check parity of "NO"
  - 8-bit
  - ASCII character set
  - If there is an option for flow control, set XON/XOFF software flow control (DC1/DC3).
- Shut down to single-user state:

  ```
  shutdown -y -i1
  ```
- Power up the system: turn on the console terminal and wait for the cursor to appear. After the cursor appears, turn on your computer, following the instructions in your computer installation manual.

17 User and Group Management

Introduction

This chapter tells you how to set up and control accounts for users and user groups, file and directory access, and command authorization on your computer. You can do any of these functions by selecting the appropriate "task" from a series of menus provided for administration. To access the "system administration" menu for user and group management, type

```
sysadm users
```

The following menu will appear on your screen:

```
1        User Login and Group Administration

add      - Adds Users or Groups
defaults - Defines Defaults for Adding Users
list     - Lists Users or Groups
modify   - Modifies Attributes of Users or Groups
password - (Re-)defines User Password Information
remove   - Removes Users or Groups
```

If you prefer not to use the menus, you can perform these tasks by executing shell-level commands, instead. The following table shows the shell commands that correspond to the tasks listed on the menu.

Task to Be Performed	sysadm Task	Shell Command
Adding a user	add	useradd(1M)
Adding a group	add	groupadd(1M)
Changing a password	password	passwd(1)
Defining defaults for adding users	defaults	useradd(1M)

Listing users and groups	list	logins(1M)
		listusers(1M)

Modifying user attributes	modify	usermod(1M)
Modifying group attributes	modify	groupmod(1M)
Removing a user	remove	userdel(1M)
Removing a group	remove	groupdel(1M)

Each task listed above is explained fully later in this chapter. In addition, the *User's Reference Manual* and the *System Administrator's Reference Manual* provide explanations for the shell commands.

Overview of User and Group Management

The job of managing the accounts of individual users and user groups that work on your computer consists of several responsibilities, the most important of which are described below.

- The most important of these responsibilities is system security and controlling access to your computer. By assigning and maintaining logins and their passwords you will control who can access your computer.

  System security can also be implemented on another level once a user logs in. Your computer uses a concept of "group membership" to control access to certain files and directories. Each file and directory is a member of a group (identified with a "permission code"). Those members belonging to the same group as the file or directory are allowed access. By establishing and maintaining user group assignments, you can control user access to specific directories and files on your system. (Details of these permissions are described later in this chapter.)

- Another responsibility of user management lies in streamlining your environment to suit the particular needs of your users. For example, you may want to organize system resources differently, depending on whether your users are programmers or writers. See the "Performance Management" chapter for information about how to contour system performance to meet these specialized user needs.

- Finally, an ongoing task associated with managing users is keeping the users on your system informed about system status and servicing schedules. For routine messages, use the `motd` (message of the day) facility, the `news` command, or the `mail` command. In emergencies, send messages with the `wall` command. These commands are all described later in this chapter.

Suggestions for User and Group Management

The scope of your responsibility for managing users and groups on your system depends on the size of your company and the size of your computer. The following are some suggestions for administering users and groups:

- If you a need to create a new group ID for new users, create the group ID before assigning the group to the users (see `groupadd`(1M) for more information).
- Inevitably, some users will forget their passwords. When this occurs, it is wise to have the user see you in person to have the password changed.
- Tell your users how to use file and directory permissions, and explain the function of `umask` in the user profile (brief written instructions usually suffice). (See "The File Creation Mask (`umask`)" section later in this chapter for an explanation of `umask`.)
- Develop a method for collecting users' problems and questions. For example, you might send all users a "hot-line" telephone number or forms they can fill out when requesting new login IDs.
- Create a skeletal home directory for all new user accounts. An example of a skeletal home directory is in `/etc/skel`. The `skel` directory contains a standard user profile (`.profile`). (See "The User's Profile" later in this chapter for more information about this directory.)

Controlling Access to the System and Data

This section describes some basic procedures for ensuring the security of your computer and the data stored in it. The first line of defense is making sure that no one may use your computer except those authorized by you. By issuing login names and passwords which, together, serve the same function as a combination lock on a safe, you can maintain control of who uses your computer system.

In addition, you can protect the data on your computer by assigning permission codes to individual directories and files. These codes restrict access to groups and individuals that you specify.

The Function of Logins

When a user requests access to a computer, the `login:` prompt is displayed and the user must enter a string of characters called a "login name." The system then searches a file for a matching string. If a match is found, the `Password:` prompt is displayed and the user types his or her password. The computer checks to make sure that the password entered is the correct one for the login name entered and, if it is, the user is allowed access to the computer.

NOTE

Some administrative operations require the use of an authorized login. This means that you must log in with a special administrative or system login name to perform this operation (see the "Security" chapter for a list of these logins).

WARNING

Assign passwords to all logins. Refer to the "Security" chapter for information about passwords and other security precautions you should take with your computer.

Login names and passwords are maintained with the commands `useradd`, `usermod`, and `userdel`, as described in the "Creating and Maintaining Accounts" section later in this chapter.

The Function of Directory and File Permissions

In addition to controlling access to your computer, you can control access to particular files and directories. Specifically, you can grant or deny access permission to use data to individual users, groups of users, or all the users on your computer by assigning the appropriate values to a "permission code" associated with the relevant directory or file.

The permission code is a ten-character string that shows who can access the data in question. It can be viewed, as part of a directory or file listing, by running the `ls` command, with the `-l` option, as shown in the following example:

```
# ls -l
-rwxrwxrwx   1 jean      doc        912 Aug 17 21:11 file1
#
```

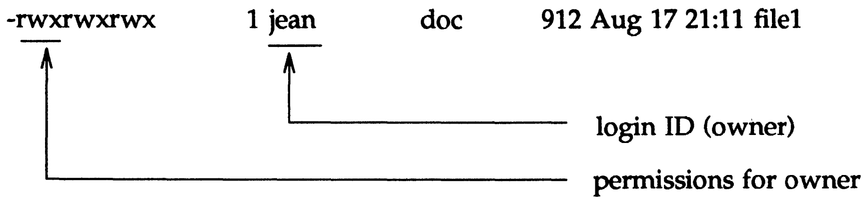
The first character in the listing shows the type of data object under consideration. Data objects include directories, ordinary files, named pipes, symbolic links, and so on.

This character is followed by a three-character string that grants (or denies) permission to access the data to the "owner" of the file, that is, the owner of the login name, who is listed in the third field (in the above example, `jean`). The owner may be the creator of the data or simply the current owner of that data. (For details about reassigning ownership of a file, see `chown(1)` in the *User's Reference Manual*.)

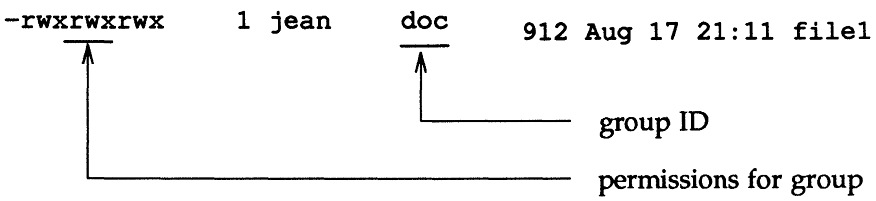
The three-character string in the code defines the following types of access permissions:

r	Permission granted to read (that is, to examine but not to change the data)
w	Permission granted to write (that is, to change the data)
x	Permission granted to execute (that is, to execute the command or shell script in the file)

If permission is denied, a hyphen (-) appears instead of the appropriate letter in the string.



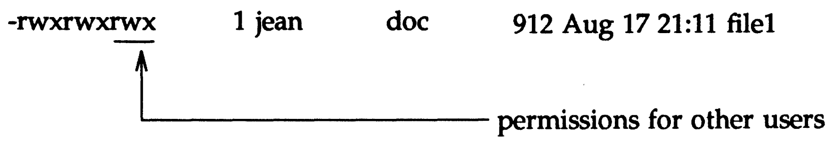
The next three-character string grants (or denies) permission to access the data to the group of users assigned the same group ID as the owner of the login name.



Group IDs allow users with common interests to share directories and files. Any file created by a member of the group carries the group ID as a secondary identification. (The primary identification is the login name of the file's owner.) By manipulating the permissions code associated with a directory or file, the owner (or anyone using the login name of the owner) can grant read, write, or execute privileges to other group members.

Once users have logged in, their access to files and directories can be controlled through the group ID component of the permission code. If a person is assigned to the same group as a file or directory (and if permissions allow), a user may access the data. If a person is not a member of this group, access can be blocked. Group accounts are administered with the `groupadd`, `groupmod`, and `groupdel` commands. These commands are described in the "Creating and Maintaining Accounts" section later in this chapter.

The last three-character string grants (or denies) permission to access the data to all other users of the computer.



```
-rwxrwxrwx    1 jean    doc    912 Aug 17 21:11 file1
```

Creating and Maintaining Accounts

Login names allow access to a computer; group IDs allow access to data. Create at least one login name for every user. Then assign each login name to one or more groups with the `groupadd` command and either the `-g` or `-G` options.

`groupadd -g`
: Assigns primary group membership; this group ID will be assigned to all directories and files created by the owner of the login name

`groupadd -G`
: Assigns supplementary group membership; this group ID will entitle the owner of the login name to access directories and files to which the same group ID has been assigned

Remember that a group ID must exist before a login can be assigned to it.

You should create group IDs (with the `groupadd` command) to restrict access to special commands or to information about a special project. Then assign the login IDs of those users who need access to the project or commands (secondary group membership) with the `groupmod` command. Remember that when creating groups, a user can be a member of several groups but a member of only one primary group. (Refer to the `groupadd(1M)` command in the *System Administrator's Reference Manual* for more information about group IDs.)

Adding a User Login

Many administrators must add new users to their systems frequently. If you have a large computer and you must add many new users every day, you may want to schedule these additions at a time of day when the system is not busy. If you have a small computer, the time needed to add one user is negligible.

The simplest way to create an account for a new user is by typing

```
useradd login_name
```

where *login_name* is the login name of that user. This command defines (but does not create) a default directory as the new user's home directory.

If you want the default home directory to be created, as well as defined, you must specify the `-m` option, as follows:

```
useradd -m login_name
```

If you want to define (but not create) a directory other than the default one as the home directory, specify the `-d` option, followed by the path of the desired directory, as follows:

```
useradd -d dir_path login_name
```

Finally, if you want to create, as well as define, a home directory other than the default, you must specify both the `-m` and `-d` options, along with the path of the desired directory, as follows:

```
useradd -m -d dir_path login_name
```

The `useradd` command creates entries for the new user in two files: `/etc/passwd` and `/etc/shadow`. These files contain user login credentials, i.e., login name, password, UID, GID, etc. (See also `password`(4).)

Optional Activities When Adding Users

Before creating an account for a new user, you may want to create new files (or edit existing ones) that can provide a particular working environment for that user. The `/etc/skel` directory contains several such files, including the `.profile` file. These files can be copied to the home directory of each new user. By doing this you can ensure that every user you add to your computer will have the same startup environment. (For details, see the description of the `-k` option below.)

NOTE: The system file entries created when you run the `useradd` command have a limit of 512 characters per line. If you specify long arguments to several options, you may exceed this limit.

The `useradd` command allows you to define many parameters for a new user account and the login name associated with it. Specifically, it lets you do the following functions (listed here by option):

`-b` *base_dir*

Define a directory as the default base directory: the directory in which accounts for new users will be created when the `-d` option is not used. The path for a user's home directory is composed of *base_dir,* followed by his or her login name. For example, suppose you define a base directory called `testgrp` as the home directory, Then you create an account (by running the `useradd` command without the `-d` option) for a user with the login name `jean`. The home directory for this user will be `/testgrp/jean`.

NOTE: You cannot create a directory by defining it as the default base directory; the directory you specify with the `-b` option must already exist. (See also the description below the `-m` option.)

`-c` *comment*

Store helpful information about the owner of each login name (such as his or her full name) in the `/etc/passwd` entry for that user. If your comment contains spaces between words, it must be enclosed in double quotes, as in the following example:

```
-c "Art and Graphics"
```

Double quotes are not needed when the comment does not include spaces, as in the following example:

```
-c Art_and_Graphics
```

`-D [options]`

When the `useradd` is invoked with only the `-D` option, the default values for *group, base_dir, skel_dir, shell, inactive,* and *expire* are displayed.

When invoked with the `-g`, `-b`, `-f`, or `-e` option (or any combination of these options), the `useradd -D` command sets the default values for the parameter shown by the option. (As installed, the default group is `other` (group ID of `1`) and the default value of *base_dir* is `/home`.) Subsequent invocations of `useradd` use these default values unless specified otherwise.

`-d` *dir*	The non-default home directory of the new user. You must specify a full path to this *home_directory* (such as `/home2/login`). See also the `-b` option.
`-e` *expire*	The date on which a login can no longer be used; after this date, no user will be able to access this login. (This option is useful for creating temporary logins.) You may type the value of the argument *expire* (which is a date) in any format you like (except a Julian date). For example, you may enter `10/6/90` or `October 6, 1990`. A value of `""` defeats the status of the expired date.
`-f` *inactive*	The maximum number of days allowed between uses of a login name before that login is declared valid. Normal values are positive integers. A value of `-1` defeats the status.
`-G` *group*	An existing group's integer ID or character-string name. It defines the new user's supplementary group membership. Supplementary groups are those that can be accessed with the `newgrp` command. (See also the `-g` option.)
`-g` *group*	An existing group's integer ID or character-string name. Without the `-D` option, it defines the new user's primary group membership and defaults to the default group. You can reset this default value by invoking `useradd -D -g` *group*.
`-k` *skel_dir*	Allows you to copy the files in a "skeleton" directory (directories and files containing programs and tools that may be useful to every new user, such as `rje` and `.profile`) into a new user's home directory. The system provides a default skeleton directory: `/etc/skel`. You cannot create a skeleton directory with this option; the directory you specify with `-k` must already exist.

`-m`	This option creates a home directory if one doesn't already exist. It takes no arguments. If a home directory already exists, it must have read, write, and execute permissions by *group*, where *group* is the user's primary group. The new login remains locked until the `passwd` command is executed (see the `-b` and `-g` options). If the `useradd` command is executed without the `-m` option, a default home directory is not created for the user. You must create a home directory and give the user ownership of it. (For instructions on giving the user ownership of a directory, see `chown`(1M).)
`-o`	Allows a UID to be duplicated (that is, non-unique). This option should be used only in special cases.
`-s` *shell*	Use a non-default program as the user's shell on login. The value of *shell* must be a valid executable file and you must specify a full path to this *program* (such as `/usr/bin/ksh`). `/sbin/sh` is the default shell when the `-s` option is not used.
`-u` *UID*	The UID of the new user. This UID must be a non-negative decimal integer below MAXUID as defined in `/usr/include/sys/param.h`, a file that contains default values for various system parameters. The UID defaults to the next available (unique) number above the highest number currently assigned. For example, if UIDs 100 and 105 are assigned, the next default UID number will be 106. (UIDs from 0-99 are reserved.)

To display the default values for the `useradd` command:

Type `useradd -D`

To set default values for the `useradd` command:

Type `useradd -D` with any or all of the following options:

- `-b` *default_base_dir*
- `-e` *default_expire_date*

- `-f` *default_inactive_days*
- `-g` *default_group*

To add a user, specifying non-default values:

1. For each user you want to add, type:

   `useradd` *option argument* `-m` *login*

   where *option* and *argument* may be any or all of the following:

option	argument
`-c`	*comment*
`-d`	*home_directory*
`-e`	*expire*
`-f`	*inactive*
`-G`	*supp_group*
`-g`	*primary_group*
`-m`	---
`-o`	---
`-s`	*shell*
`-u`	*UID*

   Specifying more than one *supp_group* is allowed. If you do so, separate individual items in the list in one of two ways: by commas, without intervening spaces (*supp_group1,supp_group2*) or with spaces, enclosing the entire list in double quotes ("*supp_group1 supp_group2*").

2. For each user you have added, you must create a temporary password. To do this, execute `passwd` *login*

3. In addition, you may want to set up some parameters for the password you are creating. For example, for security reasons, you might want to restrict the amount of time a user is allowed to use a temporary password before replacing it with a permanent one. You can create restrictions such as this at any time. Consider the following options when you create a temporary password.

   - You can force a new user to replace a current (or temporary) password with a permanent one at the next login session: `passwd -f` *login*

- You can assign a minimum number of days that the new user will be forced to keep a current (or temporary) password before being allowed to replace it: `passwd -n` *login*
- You can request a date (specified by the number of days before the expiration of the current password) on which the user will be warned about the impending expiration: `passwd -w` *login*
- You can specify the maximum number of days that a user will be allowed to keep a current password before being forced to replace it: `passwd -x` *login*

Removing a User

There are two ways that you can remove a user's login: you can remove just the login entry, or you can remove the login entry and the user's files. If you remove just the login entry, the user's files remain. Preserving these files and directories may be a good idea in case any of them are needed by other users.

If you remove the user's login from the computer with the `-r` option of the `userdel` command, all files and directories associated with this login are removed from the computer; after this step is taken, these files are accessible only from backup tapes.

To remove only the login entry:

Type

`userdel` *login*

To remove the login entry and the user's home directory:

Invoke

`userdel -r` *login*

Adding a Group

Many administrators must add new groups to their systems frequently. If you have a large computer and you must add many new groups every day, you may want to schedule these additions at a time of day when the system is not busy. If you have a small computer, the time needed to add one group is negligible.

NOTE Groups must be added to your computer before users can be assigned to them.

To add a group:

1. Type

   `groupadd` *group_name*

   and the group will be automatically assigned a GID.

   If you want to assign a specific GID to the group, include the `-g` option, as follows:

   `groupadd -g` *GID group_name*

2. Then, if you want to add users to the group, do so by executing the `useradd` command.

Renaming a Group

As projects and the needs of your users change, you may need to change the name of a group without changing group membership. To do so, type

`groupmod -n` *new_group_name old_group_name*

where *new_group_name* is the new name for the group and *old_group_name* is the current name of the group. For example, to change a group name from `tech` to `pub`, type `groupmod -n pub tech`.

Modifying User and Group Attributes

In large companies, circumstances in users' lives change regularly: users transfer departments, receive promotions, change projects, and so on. The status (attributes) of these users may need to be updated when such an event occurs. For example, changing projects may mean that access to a certain group is no longer required.

In addition, special projects may be created where access to project information must be restricted to a select few. In these cases, a new group (and possibly new login names should be created for this project. (See "Adding a User" earlier in this chapter.) The user's profiles for the login names assigned to this group should have the mask set appropriately (see "The File Creation Mask (`umask`)" later in this chapter).

The `usermod` command is used to change the status or attributes of existing user logins. This command has several options that are described below.

`-b` *base_dir*

Assigns a default base directory in which new accounts for users will be created when the `-d` option is not used. The path of a user's home directory consists of the name of the base directory, followed by that user's login name. For example, if *base_dir* is defined as `/home2`, and a new account is later created (by running `usermod` without the `-d` option) for a user with login name `jean`, the home directory for that login will be `/home2/jean`.

NOTE: The `base_dir` you specify must already exist. (See also the description below of the `-m` option.)

`-c` *comment*

Stores a text string in the user's `/etc/passwd` entry. A comment can be a short description of the login, the user's full name, and so on. You must use opening and closing double quotes for spaced words (such as `-c "Art and Graphics"`). Double quotes are not needed when spaces are not used between words (such as `-c Art_and_Graphics`).

`-d` *dir*	The new home directory of the user. You must specify a full path to this *home_directory* (such as `/home2/login`). See also the `-b` option.
`-e` *expire*	The date on which a login can no longer be used; after this date, no user will be able to access this login. (This option is useful for creating temporary logins.) You may type the value of the argument *expire* (which is a date) in any format you like (except a Julian date). For example, you may enter `10/6/90` or `October 6, 1990`. A value of `""` defeats the status of the expired date.
`-f` *inactive*	The maximum number of days allowed between uses of a login name before that login is declared valid. Normal values are positive integers. A value of `-1` defeats the status.
`-G` *group*	An existing group's integer ID or character-string name. It defines the new user's supplementary group membership. This *group* must be a non-negative decimal integer below `NGROUPS_UMAX` as defined in `/usr/include/sys/param.h`, a file that contains default values for various system parameters. Supplementary groups are those that can be accessed with the `newgrp` command. (See also the `-g` option.)
`-g` *group*	An existing group's integer ID or character-string name. It defines the new user's primary group membership and defaults to the default group.
`-l` *new_login*	A string of printable characters that specifies the new login name for the user. It may not contain a colon (`:`) or a newline (\n).
`-m`	This option has no argument and moves the user's home directory to the new location specified with the `-d` option. If this new location already exists, it must have read, write, and execute permissions by *group*, where *group* is the user's primary group.
`-o`	Allows a UID to be duplicated (that is, non-unique). This option should be used only in special cases.

`-s` *shell*	Use a non-default program as the user's shell on login. The value of *shell* must be a valid executable file and you must specify a full path to this *program* (such as `/usr/bin/ksh`). `/sbin/sh` is the default shell when the `-s` option is not used.
`-u` *UID*	The `UID` of the user. This `UID` must be a non-negative decimal integer below `MAXUID` as defined in `<param.h>`.

To modify user attributes:

1. For each user you want to add, type:

   `usermod` *option argument* `-m` *login*

   where *option* and *argument* may be any or all of the following:

option	argument
-c	*comment*
-d	*home_director*
-e	*expire*
-f	*inactive*
-G	*supp_group*
-g	*primary_group*
-l	*new_login*
-m	---
-o	---
-s	*shell*
-u	*UID*

   You can specify more than one *supp_group*. When doing so, they must be either separated by commas with no spaces as

   ```
   supp_group1,supp_group2
   ```

   or enclosed in double quotes with spaces as

   ```
   "supp_group1 supp_group2"
   ```

2. For each user you have added, invoke `passwd` *login* to assign a temporary password. Invoke the `passwd` command with the `-f`, `-n`, `-w`, or `-x` options as described below:

`-f`	Force the new user to change the password at the next login session.
`-n`	Assign the minimum number of days (if necessary) that the new user will be allowed to change his or her password.
`-w`	Set the number of days before the password expires that the user will be warned.
`-x`	Assign the maximum number of days (if necessary) that the new user will be allowed to keep his or her existing password.

To modify group attributes:

1. Invoke the `groupmod` command, with the appropriate option, to do one of the following routines (you can use one or all options at the same time):

   - Change the group ID:

     `groupmod -g` *option*

   - Override the unique group ID constraint so you can use a single group ID for more than one group:

     `groupmod -o` *option*

     Use this option only when you must; use of the same group ID for multiple groups is not recommended.

   - Change the group name:

     `groupmod -n` *option*

Listing User and Group Information

During normal administrative activities, it may be necessary to check up on user and group assignments. The following procedures allow you to do this.

To list all users and groups:

Type

```
      listusers
```

To list information about one user or more users:

Type

```
      listusers -l login1 login2
```

To list users belonging to a specific primary or secondary group:

Type

```
      listusers -g group_name_list
```

where the names of groups are separated by commas.

To list logins with duplicate login IDs:

Type

```
      logins -d
```

To list logins with supplemental group IDs:

Type

```
      logins -m
```

To list logins with unassigned passwords:

Type

```
      logins -p
```

To display extended information about logins:

Type

```
      logins -x
```

Changing Passwords

Inevitably, some users forget their passwords. You should establish a policy for changing forgotten passwords that will help you protect the security of your system.

To Change a User's Password:

1. Invoke the `passwd` command with the user's login as follows:

   `passwd` *login*

2. Type a temporary user password after the `New password:` prompt.
3. Type this temporary user password again after the `Re-enter new password:` prompt.
4. Make sure the user changes the password immediately by executing the `passwd` command with the `-f` option, as follows:

   `passwd -f` *login*

To Change Passwords for System Commands Logins:

1. Invoke the `passwd` command with the login name for the appropriate system command, as follows:

   `passwd` *system_command*

   system_command is a command to which you want to restrict access. For example, if you want to restrict access to the `sysadm` system command, type

   ```
   passwd sysadm
   ```

2. Type a new password after the `New password:` prompt.
3. Type this new password again after the `Re-enter new password:` prompt.

Streamlining the Work Environment: System and User Profiles

Your computer performs several routines between the time a user logs in and the time a shell prompt appears. These routines are listed in the two files that are executed during the login sequence: the "system profile" (`/etc/profile`) and the "user profile" (`$HOME/.profile`). This section describes these files and explains how you can change them to enhance the performance of your system so it best serves the needs of your users.

The System Profile

When a user enters a valid login name and password, the computer runs a program called the system profile (`/etc/profile`). This program is an executable ASCII file that performs several important actions:

- It defines and exports some environment variables (see the "Environment Variables" section of this chapter). Refer to the "Performance Management" chapter for additional information about environment variables.
- The message of the day is displayed (see the "Message of the Day" section of this chapter).
- A list of news items is displayed if the user is not `root` (see the "News" section of this chapter).
- A message about mail items is displayed if the user has mail (see the "Sending Electronic Mail to Users" section of this chapter).
- The default user mask is defined. (See "The File Creation Mask (`umask`)" section of this chapter.)

A sample `/etc/profile` is shown in Figure 17-1.

Figure 17-1: A Sample System Profile (/etc/profile)

```
#ident  "@(#)nsadmin:profile      1.3"
# The profile that all logins get before using their own .profile.

trap ""  2 3
export LOGNAME

. /etc/TIMEZONE

# Login and -su shells get /etc/profile services.
# -rsh is given its environment in its .profile.
case "$0" in
-su )
  export PATH

# Uncomment this script if you wish to use secure RPC facility
#
# ps -e | grep rpcbind 1>/dev/null
# if [ $? = 0 ]
# then
#    ps -e | grep keyserv 1>/dev/null
#    if [ $? = 0 ]
#    then
#        echo "Please login to the network"
#        /usr/bin/keylogin
#    else
#      echo `date`: "secure rpc nonfunctional; keyserv is down" >>/var/adm/log/rpc.log
#    fi
# else
#    echo `date`: "secure rpc nonfunctional; rpcbind is down" >>/var/adm/log/rpc.log
# fi
#
  ;;
-sh )
  export PATH

  # Allow the user to break the Message-Of-The-Day only.
  trap "trap '' 2"  2
  cat -s /etc/motd
  trap "" 2

  if mail -e
  then
     echo "you have mail"
  fi
```

(continued on next page)

Figure 17-1: A Sample System Profile (`/etc/profile`) (continued)

```
  if [ "${LOGNAME}" != "root" ]
  then
     news -n
  fi

# Uncomment this script if you wish to use secure RPC facility
#
# ps -e | grep rpcbind 1>/dev/null
# if [ $? = 0 ]
# then
#     ps -e | grep keyserv 1>/dev/null
#     if [ $? = 0 ]
#     then
#        echo "Please login to the network"
#        /usr/bin/keylogin
#     else
#      echo `date`: "secure rpc nonfunctional; keyserv is down" >>/var/adm/log/rpc.log
#     fi
# else
#      echo `date`: "secure rpc nonfunctional; rpcbind is down" >>/var/adm/log/rpc.log
# fi
#

  ;;
esac

umask 022
trap  2 3
```

You can edit the system profile to change existing definitions. For example, you may want to change the value assigned to your system mask from 022 to 077 to make files and directories created by users more secure. (See chmod(1) in the *User's Reference Manual* for information on file permissions.) Or you may want to change the default PATH to provide access to locally developed commands.

You can also edit the contents of the system profile to automate certain routines. For example, you can add shell commands to this file to display the current time and date, inform a user that he or she has mail, and tell a user the number of people logged in on the computer, as shown in Figure 17-2.

Figure 17-2: Additional Shell Script for the System Profile (`/etc/profile`)

```
echo `date`
mail -e
n=`who | wc -l`
echo "There are $n users on `uname -n`."
```

NOTE You should change the system profile only when the change you make will be beneficial to all users of the computer.

The User's Profile

For new users, System V provides a directory called `/etc/skel` that contains the standard user profile, `.profile`. If you want to provide new users with other directories and files (such as an `rje` directory or a `.mailrc` file), you can add them to the `skel` directory. If you prefer, you can create a new "skeletal" directory with a customized `.profile`. Then, whenever you add a new user login name to your system, you can provide a set of directories and files for the new user immediately by having the contents of `skel` (or your own version of it) copied into the user's home directory.

To have this copy made, specify the `-k` option with the `useradd` command, as follows:

`useradd -k` *pathname*

pathname is either `/etc/skel` or the pathname of your customized "skel" directory. This mechanism obviates the need to add directories and files, individually, to a new home directory.

After executing the system profile, the computer executes the user's profile. The user's profile executes commands and shell scripts in the same way that the system profile does. Figure 17-3 shows a sample `.profile` file.

Figure 17-3: A Sample User's Profile (`$HOME/.profile`)

```
PS1="Yes? "
HOME=/home/jean
LOGNAME=jean
PATH=:/bin:/usr/bin:/usr/lbin:$HOME/bin
CDPATH=`for i in $HOME $HOME/* $HOME/junk/* ; do if test -d $i ;\
     then echo ":$i\c" ; fi ; done`
TERMINFO=/home/jean/terminfo
TERM=630
export PS1 HOME LOGNAME PATH CDPATH TERMINFO TERM
umask 022
mail
```

An individual user can redefine some of the environment variables in the `.profile` to customize his or her environment (to suit factors such as the type of terminal being used, and the type of work being done). If you want to change the environment for your users, you can redefine the environment variables in the standard user profile (`/etc/skel/.profile`). The environment variables set in the user profile are defined below. If you want to change the environment for your users, you can modify this file to include some or all of the environment variables shown in Figure 17-3. These environment variables are defined below.

Environment Variables

This section describes the seven environment variables defined in the sample user profile shown in Figure 17-3. Refer to the "Performance Management" chapter for additional information about environment variables.

`PS1`	The user's shell-level prompt
`HOME`	The user's home directory. Scripts and system programs that need to reference the user's home directory use this environment variable to find it.
	If a user is moved to another file system, the only change in his or her profile needed to simulate the original working environment is a change in the definition of `$HOME`. For example, if a

user whose login name is `jean` is moved from the `/home` file system to the `/home2` file system, the only change needed in her user profile is a change in the definition of `$HOME` from `/home/jean` to `/home2/jean`. Once this environment variable has been changed, the user will have complete access to all the files and commands that were available in the environment defined by the previous value of `PATH`.

`LOGNAME` The user's login name. Scripts and system programs that need to reference the user's login name use this environment variable to find it.

`PATH` The list of directories and the order in which these directories are searched for a command requested by a user. This order may be important. When identically named commands exist in different locations, the first command found with that name is used. For example, in Figure 17-3, `PATH` is defined as `PATH=/bin:/usr/bin:/usr/lbin:$HOME/bin`. Therefore `/usr/bin` is defined before `/home/jean/bin`. If a user invokes a command called `sample` without specifying its full path name, and this command resides in both `/usr/bin` and `/home/jean/bin`, the version found in `/usr/bin` will be used.

`CDPATH` Specifies the directories searched when a unique directory name is typed without a full pathname. This environment variable is used as an argument to the `cd` command. For example, two directories exist under `/home/jean`: `bin` and `rje`. If you are in the `/home/jean/bin` directory and type `cd rje`, you will change directories to `/home/jean/rje` even though a full path is not specified.

`TERMINFO`

Supported terminal definitions are found in the default terminal information database, `/usr/share/lib/terminfo`. When using an unsupported terminal, a user can create a definition for it in another terminfo directory (see `tic`(1M) in the *System Administrator's Reference Manual*) and define `TERMINFO` as a path to this directory in his or her profile (such as `/home/jean/terminfo`). The system first checks the `TER-MINFO` path defined by the user. If a definition for a terminal is not found in the `TERMINFO` directory defined by the user, the computer searches the default directory, `/usr/share/lib/terminfo`, for a definition. If a definition is not found in either location, the computer identifies the terminal as "dumb." A description of the `terminfo` directory is provided in the "Directories and Files" appendix. (Refer to `tic`(1M) in the *System Administrator's Reference Manual* and to `curses`(3X), `term`(4), and `terminfo`(4) in the *Programmer's Reference Manual* for more information about the commands used with this directory.)

`TERM` The terminal used by this user. When the user invokes an editor, the computer looks for a file with the same name as the definition of this environment variable (in Figure 17-3, a 630 terminal). The system searches the directory referenced by `TERMINFO` (in Figure 17-3, `/home/jean/terminfo`) to determine the characteristics of the terminal.

New environment variables can be defined in a user's profile at any time. By convention, environment variables are defined as follows:

VARIABLE_NAME=value.

The variable name appears in uppercase, followed by an equals sign and the value. Once an environment variable is defined, it can be used globally by executing the `export` command with the environment variable as an argument to it.

The File Creation Mask (`umask`)

The default permissions (mask) used for files and directories created by a user are determined by the values assigned to the file creation mask. The `umask` command assigns values to the mask. The system profile may contain the `umask` command to establish these permissions. Default permissions may be redefined (subtracted from these default permissions) in a user's profile.

For example, if the default mask is set by the system profile to mode `027`, the command line `umask 022` causes newly-created directories to be assigned mode `755` (`drwxr-xr-x`) and newly-created files to be assigned mode `644` (`-rw-r--r--`). If the default mask is set by the system profile to mode `027`, the command line `umask 027` causes newly-created directories to be assigned mode `750` (`drwxr-x---`) and newly-created files to be assigned mode `640` (`-rw-r-----`). The system checks the permission settings before allowing or denying access to a file or directory.

NOTE The value of the executable bit is ignored for the creation of text files but not for the creation of directories.

Access can be given to one or more of the following sets of users:

- The owner of the data
- People with the same group ID as the owner of the data
- All other people using the computer

These permissions are explained in more detail in "The Function of Directory and File Permissions" earlier in this chapter. (See umask(1) in the *User's Reference Manual*.)

Communicating with Users

As an administrator, you will frequently need to send messages to the users on your computer. From time to time, you will also need to collect forms and feedback from them. This section describes how to do both.

Sending Messages to Users

You can communicate with the users on your system with any of the following four tools:

- The message of the day facility
- The news program
- The write-to-all-users facility
- The mail utilities

This section describes these four tools.

Message of the Day

The message of the day facility allows you to send announcements or inquiries to all users of the computer simply by adding the text of your messages to the `/etc/motd` file. Whenever users log in, your messages will be displayed on their terminals. A sample message of the day is shown below:

```
The system will be down from 1700 to 2300 hours on Friday,
September 30, for upgrades and preventive maintenance.
```

Edit the `/etc/motd` file regularly to remove obsolete notices.

Use the message of the day sparingly; if you inundate users with trivial messages, they will learn to ignore all your messages.

News

The `/var/news` program works like an electronic bulletin board: it uses a designated directory to hold files containing announcements that users may peruse when they like.

To post a news item for all users:

1. Create a file and type your message in that file. Give the file a name appropriate to the message it contains.
2. Move this message to the `/var/news` directory (`mv` *filename* `/var/news`).

To read news items, a user types `news` at the shell prompt. The computer will then display all files in the `/var/news` directory.

You can add the `news` command with no options to the last line of the default user profile (`/etc/skel/.profile`) so that new users added to your computer will have this command in their `.profiles`. Or you can ask users to add this same command to their own user profiles (`$HOME/.profile`). Doing so causes news items to be displayed just before a user obtains his or her shell prompt.

Unlike the message of the day, which users cannot turn off, the `news` program gives users a choice of several possible actions:

read all items	If a user types `news` with no arguments, all current news items posted since the last time the user entered this command are displayed on the user's terminal.
select some items	If a user types `news` with the names of one or more items as arguments, only those items selected are printed.
read and delete	If the DELETE key is pressed while a news item it printing, the printing stops and the next news item is started. Pressing DELETE again within one second of the first causes the program to terminate.
ignore all items	If a user is too busy to read announcements at the moment, they can be ignored. The item names are displayed each time this user logs in. The news items remain in `/var/news` until they are removed.

remove all items — If a user simply wants to remove the display of item names without looking at the items, two techniques will work.

1. Type `touch .news_time` to update the time-accessed and time-modified fields of the `.news_time` file.
2. Type `news > /dev/null` to print the news items on the null device.

See `news`(1) in the *User's Reference Manual* for more information about this command.

Write to All Users

You can write to all logged-in users via the `wall` command. For example, if the computer must be taken out of service, you can warn users with the following type of message:

```
# wall
The system is coming down in ten minutes for unexpected maintenance.
Please log off soon in order to save your files.
CTRL-d
#
```

While the `wall` command is a useful device for getting urgent information out quickly, some users dislike having messages printed on their terminals while they are working. Many users guard against such uninvited messages by including the command

```
mesg n
```

in their `.profile`. The `mesg` command with the `n` option blocks the output of the `wall` command; messages sent by ordinary users with the `wall` command will not appear on the recipient's screen. When used by the `root` login, however, the `wall` command overrides the `mesg -n` command. It is best to reserve the use of the `wall` command for those times when you need to ask users to log off from the computer.

See wall(1) in the *User's Reference Manual* for more information.

Sending Electronic Mail to Users

Two electronic mail utilities (mail and mailx) allow users to communicate with each other. If your system is connected to others by networking facilities, you can also use these utilities to communicate with people on other systems.

mail is the basic utility for sending messages. mailx uses mail to send and receive messages, but adds to it additional features that are useful for storing messages in files, adding headers, and so on.

If you use mailx, you may find it helpful to use a file called .mailrc. For details about using this file, see mailx(1) in the *User's Reference Manual.*

Collecting Users' Requests

As the system administrator for your computer, you will be asked to solve many problems. In addition to the system log described in the "Overview of System Administration" chapter, you will find it helpful to keep a log of problems reported by users. Users' problems fall into patterns; if you keep a record of how you resolve them, you will not have to start troubleshooting from scratch when a problem recurs.

Another technique that is strongly recommended is to provide users with a formal mechanism for reporting problems. The sample trouble report on the next page is an example of the type of form that can be used to keep track of system problems.

TROUBLE REPORT

Machine ______________________________

Program running ______________________________

Production or development ____________________

Type ______________________________

Symptoms ______________________________

Scope ______________________________

Error Messages ______________________________

____

____

Person reporting ____________________ Login ________

Location ______________ Phone ______________

Quick Reference to User and Group Management

The following is a summary of the user and group management activities.

- Adding a user with default options:

  `useradd` *login_ID*

- Adding a user with non-default options

  1. `useradd | -u` *user_ID* `| -g` *group_ID* `| -c` *comment* `| -d` *home_directory* `| -m | -s program` *login_name*
  2. `passwd -n` *min_days* `-x` *max_days* `-f`

- Adding a group

  `groupadd -g` *group_ID group_name*

- Listing all users and Groups

  `listusers`

- Listing user information

  `listusers -l` *login1 login2*

- Listing users assigned to a group

  `listusers -g` *group_name_list*

  where the names of groups are separated by commas.

- Listing users with duplicate user IDs:

  `logins -d`

- Listing users with supplemental group IDs:

  `logins -m`

- Listing users with unassigned passwords:

  `logins -p`

A. DEVICE NAMES AND DEFAULT PARTITIONS

A. DEVICE NAMES AND DEFAULT PARTITIONS

A Device Names and Default Partitions

Introduction

This appendix describes the device names and default partitioning information for UNIX System V Release 4.0 on the AT&T 3B2 Computer.

Device Names

Before Release 2.0

The standard disk names used to identify the integral diskette and integral hard disk drives before Release 2.0 are `/dev/idsk00` through `/dev/idsk07` for the hard disk and `/dev/ifdsk00` through `/dev/ifdsk07` for the floppy diskette. The numbers 00 through 07 identify the disk partitions.

The pre-Release 2.0 disk names are linked to the Release 4.0 disk names. These links provide a transition to the new disk names.

Release 2.0 and Later

Standard device names are used to identify the diskette, hard disk, and cartridge tape drives. The term "integral" is used to define devices driven by a controller on the system board. The first diskette drive and the first two hard disk drives are integral devices by this definition.

All disk device files use `/dev/dsk` for the block device, and `/dev/rdsk` for the raw (character) device.

- For the integral diskette drive, the controller/drive designator is `c0d0` for both the raw and block devices. For the second diskette drive, the controller/drive designator is `c`*n*`d1` for both the raw and block devices, where *n* equals the slot of the `ctc` board plus 1. The second diskette drive is driven by the first cartridge tape controller board.
- For the integral hard disk drives, the controller/drive designators are `c1d0` and `c1d1` for both the raw and block devices.
- For the first cartridge tape drive, the controller/drive designator is `c`*n*`d1` for both the raw and block device, where *n* equals the slot of the `ctc` board plus 1.

The partition or section designator is appended to the drive control/drive designator.

Release 2.0 and later releases expand the number of possible hard disk partitions from 8 to 16 (0 through f, hexadecimal). The other devices remain at a maximum of 8 partitions (0 through 7). The partitioning for the diskette and cartridge tape are fixed by the processes used to format the media. The hard disk partitioning can be configured for the best possible support of the system application.

Certain of the device partitions are used for specific functions. For example:

- Partition 0 is used for root (/).
- Partition 1 is used for swap when the device is configured as the root device for the UNIX operating system.
- Partition 2 is used for the `/usr` file system.
- Partition 3 is used for the `/stand` file system.
- Partition 6 specifies the entire device.
- Partition 7 specifies the boot partition of a device.
- Partition 8 is used for the `/var` file system.
- Partition 9 is used as the first hard disk area for user login directories (the `/home` file system).
- Partition a is used as the second hard disk area for user login directories (the `/home2` file system).

Device partitions should fall on cylinder boundaries to obtain the best possible file system performance. Boot and swap partitions (partitions 7 and 1) are special in this regard. The number of blocks assigned to the boot and swap partitions are collectively chosen to cause the next partition values to fall on cylinder boundaries. (The next partitions are normally used as file systems.) This approach eliminates wasted space that would result from strict assignment of partition values based on a modulo cylinder size for each partition. The otherwise wasted space in the boot partition is used in the swap area without degrading the system performance.

Figure A-1 describes typical 3B2 Computer controller/drive/section equipment configurations; these device names identify section 6 for the various drives. Other sections on the drives are referred to by different values for the section number, as above.

Figure A-1: Examples of Typical Device Partitions

Device	Controller	Drive	Section	Description
`c0d0s6`	0	0	6	Specifies the section 6 (s6) of the integral floppy disk drive (d0) connected to controller 0 (c0).
`c1d0s6`	1	0	6	Specifies the section 6 (s6) of the first integral hard disk drive (d0) connected to controller 1 (c1).
`c1d1s6`	1	1	6	Specifies the section 6 (s6) of the second integral hard disk drive (d1) connected to controller 1 (c1).
`c7d0s6`	4	0	6	Specifies the section 6 (s6) of the first cartridge tape drive (d0) connected to controller 7 (c7).

For more information, refer to the "Storage Device Management" chapter, and the `prtvtoc`(1M), `fmthard`(1M), `ctcfmt`(1M), and `fmtflop`(1M) pages in the *System Administrator's Reference Manual.*

Hard Disk Default Partitions

The following tables show the recommended hard disk partitions and sizes for various hard disk configurations.

Hard disks are formatted either during installation of the operating system or by the `fmthard` command. In some cases, the recommendations shown below differ from the default behavior of either or both of these methods. (These differences are pointed out in footnotes to the charts.) The information shown is for AT&T 3B2 Computers.

This information is provided as a guideline to follow should you choose not to use default partitioning on a disk, and as a means of determining if the default partitioning used by the system is adequate for your site's particular needs.

Note that all sizes are given in 512-byte blocks. The size of a 72MB disk is assumed to be 149526 blocks; the size of a 32MB disk is assumed to be 62550 blocks.

1 72MB Disk		
Partition	Use	Default Size
c1d0s0	root (/)	17010
c1d0s1	swap	14156
c1d0s2	usr	96066
c1d0s3	stand	5508
c1d0s6	entire disk	149526
c1d0s7	boot	100
c1d0s8	var	10044
c1d0s9	home	6642

2 72MB Disks		
Partition	**Use**	**Default Size**
c1d0s0	root (/)	18144
c1d0s1	swap	14156
c1d0s3	stand	5500
c1d0s6	entire disk	149526
c1d0s7	boot	100
c1d0sa	home2	42344
c1d1s2	usr	97038
c1d1s6	entire disk	149526
c1d1s7	boot	100
c1d1s8	var	10044
c1d1s9	home	111618

1 72MB Disk and 1 32MB Disk		
Partition	**Use**	**Default Size**
c1d0s0	root (/)	18144
c1d0s1	swap	14156
c1d0s2†	usr	97038
c1d0s3	stand	5500
c1d0s6	entire disk	149526
c1d0s7	boot	100
c1d0s9	home	14688
c1d1s6	entire disk	62550
c1d1s7	boot	100
c1d1s8	var	10044
c1d1sa	home2	52406

† By default, the `fmthard` command places `/usr` on the second disk; this is probably not desirable if your second disk is less than or equal to 32MB.

The installation script places `/usr` on the first disk by default when the second disk is less than or equal to 32MB.

1 32MB Disk and 1 72MB Disk††		
Partition	**Use**	**Default Size**
c1d0s0	root (/)	18144
c1d0s1	swap	14156
c1d0s3	stand	5500
c1d0s6	entire disk	62550
c1d0s7	boot	100
c1d0s9	home	24750
c1d1s2	usr	97038
c1d1s6	entire disk	149526
c1d1s7	boot	100
c1d1s8	var	10044
c1d1sa	home2	42344

†† The installation script expects the first disk to be 72MB for default partitioning; if the first disk on your machine is less than 72MB, do not use default partitioning. Instead, use the values in this chart.

Cartridge Tape Partitions

Figure A-2 defines the partition, use, size, and number of blocks for the various controller (c), drive (d), and section (s) identifiers for the 23MB cartridge tape. The `ctcfmt` command is used to format cartridge tapes (see `ctcfmt`(1M)). Note the following:

- The use of `/dev/SA/ctape1` is preferred to controller-drive-section identifiers. If you do use controller-drive-section identifiers, it is recommended that you write to the tape only on sections `3` or `6`.
- The rotational gap (Gap) and blocks per cylinder (Cyl) are defined for the device.
- Note that Volume Table Of Contents (VTOC) partitioning is fixed for the cartridge tape by the tape formatting process.
- These identifiers are applicable to both the raw and block devices.

Figure A-2: Cartridge Tape Default Partitioning

Configuration	Partition	Use	Sector Start	Size*
Cipher, 23-Megabyte Cartridge Tape, (Gap=1), (Cyl=31)	c7d0s3	usr	2	45539
	c7d0s6	entire tape	0	45541

*Size is in 512-byte blocks

Diskette Partitions

Figure A-3 defines diskette partitions in terms of use, starting sector, and total number of blocks for the various controller (c), drive (d), and section (s) identifiers for the diskette. The `fmtflop` command is used to format floppy disks (see `fmtflop`(1M)). Note the following:

- Raw and block device partitions for the entire diskette (partition 6) are linked to `/dev/rSA/diskette1` and `/dev/SA/diskette1`, respectively. To avoid accidentally writing to a different device or partition, use these names (rather than the controller, drive, and section identifiers) when specifying the entire diskette. It is recommended that only section 5 or 6 be used in controller-drive-section identifiers for the diskette drive.
- These identifiers are applicable to both the raw and block devices.
- Volume Table Of Contents (VTOC) partitioning is not applicable to the diskette drive.

Figure A-3: Floppy Disk Drive Partitions

Disk Partition	Use	Sector Start	Total Sectors*
c0d0s5	usr	1(1*18)	1404
c0d0s6	entire disk	0	1422

*Sectors are equivalent to 512-byte blocks

B Directories and Files

Overview

This appendix describes:

- Directories and files that are important for administering a system
- Directories that are new for this software release
- The reorganization of the directory structure introduced in this release
- The new organization of the root file system, and significant directories mounted on root

WARNING

To maintain a secure environment, do not change the file or directory permissions from those assigned at the time of installation.

Directory and File Relocations

For this software release, many commands and directories have been relocated. This section lists the commands that have been moved, the locations of these commands in UNIX System V Release 4, and the locations of the same commands in earlier releases of the UNIX system. UNIX System V Release 4.0 provides symbolic links between the old and new locations. However, in future software releases, these links may be removed. The asterisk (*) means that all files in the directory indicated have been moved to the new location.

Pre-Release 4 Locations	Release 4 Location
/bin/*	/usr/bin/*
/etc/bcheckrc	/sbin/bcheckrc
/etc/chroot	/usr/sbin/chroot
/etc/ckbupscd	/usr/sbin/ckbupscd
/etc/crash	/usr/sbin/crash
/etc/cron	/usr/sbin/cron
/etc/dcopy	/usr/sbin/dcopy
/etc/devnm	/usr/sbin/devnm
/etc/dfsck	/usr/sbin/dfsck
/etc/disks	/sbin/disks
/etc/drvinstall	/usr/sbin/drvinstall
/etc/editsa	/usr/sbin/editsa
/etc/edittbl	/usr/sbin/edittbl
/etc/errdump	/usr/sbin/errdump
/etc/ff	/usr/sbin/ff
/etc/finc	/usr/sbin/finc
/etc/fmtflop	/usr/sbin/fmtflop
/etc/fmthard	/sbin/fmthard
/etc/frec	/usr/sbin/frec
/etc/fsck	/sbin/fsck
/etc/fsdb	/sbin/fsdb
/etc/fsstat	/sbin/fsstat
/etc/fstyp	/sbin/fstyp
/etc/fuser	/usr/sbin/fuser
/etc/getmajor	/usr/sbin/getmajor
/etc/getty	/usr/lib/saf/ttymon
/etc/grpck	/usr/sbin/grpck

Pre-Release 4 Locations	Release 4 Location
/etc/hdeadd	/usr/sbin/hdeadd
/etc/hdefix	/sbin/hdefix
/etc/hdelogger	/usr/sbin/hdelogger
/etc/init	/sbin/init
/etc/install	/usr/sbin/install
/etc/killall	/usr/sbin/killall
/etc/labelit	/sbin/labelit
/etc/ldsysdump	/usr/sbin/ldsysdump
/etc/led	/sbin/led
/etc/link	/usr/sbin/link
/etc/log/*	/var/adm/log/*
/etc/mkboot	/usr/sbin/mkboot
/etc/mkfs	/sbin/mkfs
/etc/mknod	/sbin/mknod
/etc/mount	/sbin/mount
/etc/mountall	/sbin/mountall
/etc/mvdir	/usr/sbin/mvdir
/etc/ncheck	/usr/sbin/ncheck
/etc/npump	/sbin/npump
/etc/ports	/sbin/ports
/etc/prfdc	/usr/sbin/prfdc
/etc/prfld	/usr/sbin/prfld
/etc/prfpr	/usr/sbin/prfpr
/etc/prfsnap	/usr/sbin/prfsnap
/etc/prfstat	/usr/sbin/prfstat
/etc/prtconf	/usr/sbin/prtconf
/etc/prtvtoc	/sbin/prtvtoc
/etc/pump	/sbin/pump
/etc/pwck	/usr/sbin/pwck
/etc/rc0	/sbin/rc0
/etc/rc1	/sbin/rc1
/etc/rc2	/sbin/rc2
/etc/rc3	/sbin/rc3
/etc/rc6	/sbin/rc6
/etc/rmount	/usr/sbin/rmount

Pre-Release 4 Locations	Release 4 Location
/etc/rmountall	/usr/sbin/rmountall
/etc/rumountall	/usr/sbin/rumountall
/etc/savecpio	/usr/sbin/savecpio
/etc/setclk	/sbin/setclk
/etc/setmnt	/sbin/setmnt
/etc/shutdown	/sbin/shutdown
/etc/swap	/usr/sbin/swap
/etc/sysdef	/usr/sbin/sysdef
/etc/system	/stand/system
/etc/telinit	/sbin/init
/etc/termcap	/usr/share/lib/termcap
/etc/uadmin	/sbin/uadmin
/etc/umount	/sbin/umount
/etc/umountall	/sbin/umountall
/etc/unlink	/usr/sbin/unlink
/etc/utmp	/var/adm/utmp
/etc/volcopy	/usr/sbin/volcopy
/etc/wall	/usr/sbin/wall
/etc/whodo	/usr/sbin/whodo
/etc/wtmp	/var/adm/wtmp
/lib/*	/usr/lib/*
/shlib/*	/usr/lib/*
/unix	/stand/unix
/usr/adm/*	/var/adm/*
/usr/bin/fumount	/usr/sbin/fumount
/usr/bin/fusage	/usr/sbin/fusage
/usr/bin/mountfsys	/usr/sbin/mountfsys
/usr/bin/nlsadmin	/usr/sbin/nlsadmin
/usr/bin/powerdown	/usr/sbin/powerdown
/usr/bin/sadp	/usr/sbin/sadp
/usr/bin/strace	/usr/sbin/strace

Pre-Release 4 Locations	Release 4 Location
/usr/bin/strclean	/usr/sbin/strclean
/usr/bin/strerr	/usr/sbin/strerr
/usr/bin/umountfsys	/usr/sbin/umountfsys
/usr/lib/cron/.proto	/etc/cron.d/.proto
/usr/lib/cron/at.allow	/etc/cron.d/at.allow
/usr/lib/cron/cron.allow	/etc/cron.d/cron.allow
/usr/lib/cron/logchecker	/etc/cron.d/logchecker
/usr/lib/cron/queuedefs	/etc/cron.d/queuedefs
/usr/lib/font/*	/usr/share/lib/font/*
/usr/lib/lex/*	/usr/ccs/lib/lex/*
/usr/lib/macros/*	/usr/share/lib/macros/*
/usr/lib/spell/spellhist	/var/adm/spellhist
/usr/lib/spell/compress	/usr/share/lib/spell/compress
/usr/lib/spell/hlista	/usr/share/lib/spell/hlista
/usr/lib/spell/hlistb	/usr/share/lib/spell/hlistb
/usr/lib/spell/hstop	/usr/share/lib/spell/hstop
/usr/share/lib/terminfo/*	/usr/share/lib/terminfo/*
/usr/lib/tmac/*	/usr/share/lib/tmac/*
/usr/lib/uucp/Devconfig	/etc/uucp/Devconfig
/usr/lib/uucp/Devices	/etc/uucp/Devices
/usr/lib/uucp/Dialcodes	/etc/uucp/Dialcodes
/usr/lib/uucp/Dialers	/etc/uucp/Dialers
/usr/lib/uucp/Permissions	/etc/uucp/Permissions
/usr/lib/uucp/Poll	/etc/uucp/Poll
/usr/lib/uucp/Sysfiles	/etc/uucp/Sysfiles
/usr/lib/uucp/Systems	/etc/uucp/Systems
/usr/mail/*	/var/mail/*
/usr/man/*	/usr/share/man/*
/usr/net/nls/dbfconv	/usr/lib/saf/dbfconv
/usr/net/nls/listen	/usr/lib/saf/listen
/usr/nserve/*	/etc/rfs/*
/usr/nserve/nserve	/usr/lib/rfs/nserve
/usr/nserve/rfudaemon	/usr/lib/rfs/rfudaemon
/usr/nserve/TPnserve	/usr/lib/rfs/TPnserve

Pre-Release 4 Locations	Release 4 Location
/usr/pub/*	/usr/share/lib/*
/usr/spool/*	/var/spool/*
/usr/tmp/*	/var/tmp/*

There are some additional directories in root that did not appear in previous software releases. These directories are:

```
/export   /opt    /sbin   /stand   /var
/home     /proc
```

The root directories are explained in the next section. Important administrative files and subdirectories are explained later.

Directories in `root`

The / (root) file system contains executables and other files necessary to boot and run the system. The directories of the root file system are explained next.

/bck

The /bck directory is used to mount a backup file system for restoring files.

/boot

The /boot directory contains configurable object files created by the /usr/sbin/mkboot program (see mkboot(1M)).

/config

The /config directory contains files needed and produced by the user-level configuration program cunix (see cunix(1M)).

/dev

The /dev directory contains block and character special files that are usually associated with hardware devices or STREAMS drivers.

/dgn

The /dgn directory contains diagnostic programs.

/etc

The /etc directory contains machine-specific configuration files and system administration databases.

/export

The /export directory contains the default root of the exported file system tree.

/home

The /home directory contains user directories.

/install

The /install directory is used by the sysadm command to mount utilities packages for installation and removal (/install file system).

/lost+found

The /lost+found directory is used by fsck to save disconnected files and directories.

/mnt

The /mnt directory is used to mount file systems for temporary use.

/opt

The /opt directory is the mount point from which add-on application packages are installed.

/proc

The /proc directory is the mount point of the proc file system which provides information on the system's processes.

/save

The /save directory is used by the sysadm command for saving data on floppy diskettes.

/sbin

The /sbin directory contains executables used in the booting process and in manual recovery from a system failure.

/stand

The /stand directory is used as the mount point for the boot file system, which contains the standalone (bootable) programs and data files necessary for the system boot procedure.

/tmp

The /tmp directory contains temporary files.

/usr

The /usr directory is the mount point of the usr file system.

/var

The /var directory is the mount point of the var file system. It contains those files and directories that vary from machine to machine, such as tmp, spool, and mail. The /var file system also contains administrative directories such as /var/adm and /var/opt, the latter of which is installed by application packages.

Directories in `/etc`

This section describes the directories under the `/etc` directory, which contain machine-specific configuration files and system administration databases.

`/etc/bkup`

This directory contains machine-specific files and directories for backup and restore operations. Also contained here are files and directories that allow restore operations to be performed from single-user mode (system state 1).

`/etc/bkup/method`

This directory contains files that describe all backup and restore methods currently used on your computer.

`/etc/cron.d`

This directory contains administrative files for controlling and monitoring `cron` activities.

`/etc/default`

This directory contains files that assign default values to certain system parameters.

`/etc/init.d`

This directory contains executable files used in upward and downward transitions to all system states. These files are linked to files beginning with `S` (start) or `K` (stop) in `/etc/rc`*n*`.d`, where *n* is the appropriate system state. Files are executed from the `/etc/rc`*n*`.d` directories.

/etc/lp

This directory contains the configuration files and interface programs for the LP print service.

/etc/mail

This directory contains files used in administering the electronic mail system.

/etc/mail/lists

This directory contains files, each of which contains a mail alias. The name of each file is the name of the mail alias that it contains. (See the `mailx`(1) command for a description of the mail alias format.)

/etc/master.d

This directory contains files that define the configuration of hardware devices, software drivers, system parameters, and aliases. The files are used by `/usr/sbin/mkboot` to obtain device information for the generation of device driver and configurable module files. The `/usr/sbin/sysdef` program uses the `master.d` files to get the names of supported devices. The first step in reconfiguring the system to run with different tunable parameters is to edit the appropriate files in the `/etc/master.d` directory. (See `master`(4) in the *System Administrator's Reference Manual.*)

/etc/rc.d

This directory contains executable files that perform the various functions needed to initialize the system to system state 2. The files are executed when `/usr/sbin/rc2` is run. (Files contained in this directory before UNIX System V Release 3.0 were moved to `/etc/rc2.d`. This directory is maintained only for compatibility reasons.)

/etc/rc0.d

This directory contains files executed by /usr/sbin/rc0 for transitions to system states 0, 5, and 6. Files in this directory are linked from the /etc/init.d directory, and begin with either a K or an S. K shows processes that are stopped, and S shows processes that are started when entering system states 0, 5, or 6.

/etc/rc1.d

This directory contains files executed by /usr/sbin/rc1 for transitions to system state 1. Files in this directory are linked from the /etc/init.d directory, and begin with either a K or an S. K shows processes that should be stopped, and S shows processes that should be started when entering system state 1.

/etc/rc2.d

This directory contains files executed by /usr/sbin/rc2 for transitions to system state 2. Files in this directory are linked from the /etc/init.d directory, and begin with either a K or an S. K shows processes that should be stopped, and S shows processes that should be started when entering system state 2.

/etc/rc3.d

This directory contains files executed by /usr/sbin/rc3 for transitions to system state 3 (multi-user mode). Files in this directory are linked from the /etc/init.d directory, and begin with either a K or an S. K shows processes that should be stopped, and S shows processes that should be started when entering system state 3.

/etc/saf

This directory contains files and subdirectories used by the Service Access Facility. The following commands in /usr/sbin use /etc/saf subdirectories for data storage and retrieval: nlsadmin, pmadm, and sacadm. The following files are included:

_sactab	A list of port monitors to be started by the Service Access Controller (SAC). Each port monitor listed in this table has a _pmtab file in the /etc/saf/*pmtag* directory, where *pmtag* is the tag of this port monitor (such as /etc/saf/starlan for the starlan port monitor).
_sysconfig	The configuration script used to modify the environment for the Service Access Facility.

/etc/save.d

This directory contains files used by the sysadm command for backing up data on floppy diskettes. The following files are included:

except	A list of the directories and files that should not be copied as part of a backup is maintained in this file.
timestamp/ . . .	The date and time of the last backup (volume or incremental) is maintained for each file system in the /etc/save.d/timestamp directory.

/etc/shutdown.d

This directory is maintained only for compatibility reasons. The files contained in this directory prior to UNIX System V Release 3.0 were executable files that invoked the various functions required during the transition to the single-user mode (system states 1, s, or S). These files are now located in /etc/rc0.d.

Files in /etc

The following files are used in machine-specific configuration and system administration databases.

/etc/bkup/bkexcept.tab

This file contains a list of files to be excluded from an incremental backup.

/etc/bkup/bkhist.tab

This file contains information about the success of all backup attempts.

/etc/bkup/bkreg.tab

This file contains instructions to the system for performing backup operations on your computer.

/etc/bkup/bkstatus.tab

This file contains the status of backup operations currently taking place.

/etc/bkup/rsmethod.tab

This file contains descriptions of the types of objects that may be restored using the full or partial restore method.

/etc/bkup/rsnotify.tab

This file contains the electronic mail address of the operator to be notified whenever restore requests require operator intervention.

/etc/bkup/rsstatus.tab

This file contains a list of all restore requests made by users of your computer.

/etc/bkup/rsstrat.tab

This file specifies a strategy for selecting archives when handling restore requests. In completing restore operations for these requests, the backup history log is used to navigate through the backup tape to find the desired files and or directories.

/etc/boot_tab

This file contains a list of the file systems mounted during the configuring of a new bootable operating system (system configuration). It is used by the `/sbin/buildsys` script, along with the `/etc/vfstab` file, to mount necessary file systems. You should not need to change this file.

/etc/d_passwd

This file contains a list of programs that will require dial-up passwords when run from `login`. Each line in the file is formatted as

program:*encrypted_password*:

where *program* is the full path to any programs into which a user can log in and run. The password referred to in the *encrypted_password* is the one that will be used by the dial-up password program. This password must be entered before the user is given the login prompt. It is used in conjunction with the file `/etc/dialups`.

/etc/default/cron

This file contains the default status (enable or disable) for the CRONLOG operation.

/etc/default/login

This file may contain the following parameters that define a user's login environment:

ALTSHELL	Alternate shell status available to users (yes or no).
CONSOLE	Root login allowed only at the console terminal.
HZ	Number of clock ticks per second.
IDLEWEEKS	Number of weeks a password may remain unchanged before the user is denied access to the system.
PASSREQ	Password requirement on logins (yes or no).
PATH	User's default PATH.
SUPATH	Root's default PATH.
TIMEOUT	Number of seconds allowed for logging in before a timeout occurs.
TIMEZONE	Time zone used within the user's environment.
ULIMIT	File size limit (ulimit).
UMASK	User's value for umask.

/etc/default/passwd

This file contains the following information about the length and aging of user passwords:

MINWEEKS	Minimum number of weeks before a password can be changed.
MAXWEEKS	Maximum number of weeks a password can be unchanged.
PASSLENGTH	Minimum number of characters in a password.
WARNWEEKS	Number of weeks before a password expires that the user is to be warned.

/etc/default/su

This file contains values for the following parameters affecting the work of super users:

SULOG	A pathname that identifies a file in which a log of all su attempts may be created.
CONSOLE	Pathnames of the console on which are broadcast messages notifying you whenever someone attempts to su root.
PATH	PATH used for su users.
SUPATH	PATH used for su root users.

/etc/device.tab

This file is the device table. It lists the device alias, path to the vnode, and special attributes of every device connected to the computer.

/etc/devlock.tab

This file is created at run time and lists the reserved (locked) devices. Device reservations do not remain intact across system reboots.

/etc/saf/*pmtag*/_config

This file contains a configuration script used to customize the environment for the port monitor tagged as *pmtag* (such as /etc/saf/starlan/_config for the starlan port monitor). Port monitor configuration scripts are optional.

/etc/dgroup.tab

This file lists the group or groups to which a device belongs.

/etc/dialups

This file contains a list of terminal devices that cannot be accessed without a dial-up password. It is used in conjunction with the file /etc/d_passwd.

/etc/group

This file describes each user group to the system. An entry is added for each new group with the groupadd command.

/etc/inittab

This file contains instructions for the /sbin/init command. The instructions define the processes created or stopped for each initialization state. Initialization states are called system states or run states. By convention, system state 1 (or S or s) is single-user mode; system states 2 and 3 are multi-user modes. The "Machine Management" chapter summarizes the various system states and describes their uses. (See inittab(4) in the *System Administrator's Reference Manual* for additional information.)

/etc/mail/mailcnfg

This file permits per-site customizing of the mail subsystem. See the mailcnfg(4) manual page in the *System Administrator's Reference Manual* and "Administering the Mail Subsystem" in this guide.

/etc/mail/mailsurr

This file lists actions to be taken when mail containing particular patterns is processed by mail. This can include routing translations and logging. See the mailsurr(4) manual page in the *System Administrator's Reference Manual* and the "Mail Subsystem Administration" appendix in this guide.

/etc/mail/mailx.rc

This file contains defaults for the mailx program. It may be added by the system administrator. See mailx(1).

/etc/mail/notify and /etc/mail/notify.sys

These files are used by the notify program to determine the location of users in a networked environment and to establish systems to use in case of file error.

/etc/motd

This file contains the message of the day. The message of the day is displayed on a user's screen after that user has successfully logged in. (The commands that produce this output on the screen are in the /etc/profile file.) This message should be kept short and to the point. The /var/news files should be used for lengthy messages.

/etc/passwd

This file identifies each user to the system. An entry is automatically added for each new user with the useradd command, removed with the userdel command, and modified with the usermod command.

/etc/profile

This file contains the default profile for all users. The standard (default) environment for all users is established by the instructions in the /etc/profile file. The system administrator can change this file to set options for the root login. For example, the six lines of code shown in Figure B-1 can be added to the /etc/profile. This code defines the erase character, automatically identifies the terminal type, and sets the TERM variable when the login ID is root.

Figure B-1: Excerpt from /etc/profile

```
if [ ${LOGNAME} = root ]
    then
        stty echoe
        echo "Terminal: 5           export TERM
    fi
```

/etc/rfs/rmnttab

This file is created by the `rmount`(1M) command. This file contains a listing of unsuccessfully mounted resources or disconnected resources. These resources are polled by the `rmnttry`(1M) `cron` entry.

/etc/dfs/dfstab

This file specifies the Remote File Sharing resources from your machine that are automatically shared to remote machines when entering RFS mode (system state 3). Each entry in this file should be a `share`(1M) command line.

/etc/saf/*pmtag*/_pmtab

This is the administrative file for the port monitor tagged as *pmtag*. It contains an entry for each service available through the *pmtag* port monitor.

/etc/saf/_sactab

This file contains information about all port monitors for which the Service Access Controller (SAC) is responsible.

/etc/saf/_sysconfig

This file contains a configuration script to customize the environments for all port monitors on the system. This per-system configuration file is optional.

/etc/TIMEZONE

This file sets the time zone shell variable `TZ`. The `TZ` variable is initially established for the system via the `sysadm setup` command. The `TZ` variable in the `TIMEZONE` file is changed by the `sysadm timezone` command. The `TZ` variable can be redefined on a user (login) basis by setting the variable in the associated `.profile`. The `TIMEZONE` file is executed by `/usr/sbin/rc2`. (See `timezone`(4) in the *System Administrator's Reference Manual* for more information.)

/etc/ttydefs

This file contains information used by ttymon port monitor to set the terminal modes and baud rate for a TTY port. (See the "Service Access" chapter in this guide for more information.)

/etc/vfstab

This file provides default values for file systems and remote resources. The following information can be stored in this file:

- The block and character devices on which file systems reside

- The resource name
- The location where a file system is usually mounted
- The file system type
- Information on special mounting procedures

These defaults do not override command line arguments that have been entered manually. (See `mountall`(1M) in the *System Administrator's Reference Manual* for additional information.) Figure B-2 shows a sample of this file.

Figure B-2: Sample `/etc/vfstab` File

```
#special           fsckdev           mountp   fstype fsckpass automnt mntflags
/dev/SA/diskette1 /dev/rdiskette    /install s5     -        no      -
/dev/diskette     /dev/rdiskette    /install s5     -        no      -
/dev/dsk/c1d0s3   /dev/rdsk/c1d0s3  /stand   bfs    1        yes     -
/dev/dsk/c1d0s8   /dev/rdsk/c1d0s8  /usr2    s5     1        yes     -
/dev/dsk/c1d1s2   /dev/rdsk/c1d1s2  /usr     s5     1        yes     -
/dev/dsk/c1d1s8   /dev/rdsk/c1d1s8  /home    s5     1        yes     -
/dev/root         /dev/root         -        s5     -        no      -
/proc             -                 /proc    proc   -        no      -
```

Directories in `/usr`

This section describes the directories in the `/usr` file system. The `/usr` file system contains architecture-dependent and architecture-independent files and system administration databases that can be shared.

`/usr/bin`

This directory contains public commands and system utilities.

`/usr/include`

This directory contains public header files for C programs.

`/usr/lib`

This directory contains public libraries, daemons, and architecture dependent databases.

`/usr/lib/lp`

This directory contains the directories and files used in processing requests to the LP print service.

`/usr/lib/mail`

This directory contains directories and files used in processing mail.

/usr/lib/mail/surrcmd

This directory contains programs necessary for mail surrogate processing.

/usr/sadm/bkup

This directory contains executables for the backup and restore services.

/usr/sbin

This directory contains executables used for system administration.

/usr/share

This directory contains architecture independent files that can be shared.

/usr/share/lib

This directory contains architecture independent databases.

/usr/sadm/skel

This directory contains the files and directories built when using the `useradd` command with the `-m` argument. All directories and files under this location are built under the `$HOME` location for the new user.

/usr/ucb

This directory contains binaries from the BSD Compatibility Package.

/usr/ucbinclude

This directory contains header files from the BSD Compatibility Package.

/usr/ucblib

This directory contains libraries from the BSD Compatibility Package.

Files in /usr

This section describes the files in the /usr directories, which contain architecture-dependent and architecture-independent files and system administrative databases that can be shared.

/usr/sbin/rc0

This file contains a shell script executed by /usr/sbin/shutdown for transitions to single-user state, and by /sbin/init for transitions to system states 0, 5, and 6. Files in the /etc/shutdown.d and /etc/rc0.d directories are executed when /usr/sbin/rc0 is run. The file K00ANNOUNCE in /etc/rc0.d prints the message `System services are now being stopped.` Any task that you want executed when the system is taken to system states 0, s, 5, or 6 is done by adding a file to the /etc/rc0.d directory.

/usr/sbin/rc1

This file contains a shell script executed by /sbin/init for transitions to system state 1 (single-user state). Executable files in the /etc/rc.d directory and any executable files beginning with S or K in the /etc/rc1.d directories are executed when /usr/sbin/rc1 is run. All files in rc1.d are linked from files in the /etc/init.d directory. Other files may be added to the /etc/rc1.d directory as a function of adding hardware or software to the system.

/usr/sbin/rc2

This file contains a shell script executed by /sbin/init for transitions to system state 2 (multi-user state). Executable files in the /etc/rc.d directory and any executable files beginning with S or K in the /etc/rc2.d directories are executed when /usr/sbin/rc2 is run. All files in rc2.d are linked from files in the /etc/init.d directory. Other files may be added to the /etc/rc2.d directory as a function of adding hardware or software to the system.

/usr/sbin/rc3

This file is executed by /sbin/init. It executes the shell scripts in /etc/rc3.d for transitions to RFS mode (system state 3).

/usr/sbin/rc6

This shell script is run for transitions to system state 6 (for example, using shutdown -i6). If the operating system needs to be reconfigured, the /sbin/buildsys script is run, and, if the reconfiguration is successful, /usr/sbin/rc6 reboots the operating system without running diagnostics. If the reconfiguration is unsuccessful, a shell is spawned.

/usr/sbin/shutdown

This file contains a shell script to shut down the system gracefully in preparation for a system backup or scheduled downtime. After stopping all nonessential processes, the shutdown script executes files in the /etc/shutdown.d directory by calling /usr/sbin/rc0 for transitions to system state 1 (single-user state). For transitions to other system states, the shutdown script calls /sbin/init.

/usr/share/lib/mailx/mailx.help and /usr/share/lib/mailx/mailx.help.

Help files for mailx. The file mailx.help.~ contains help messages for mailx's tilde commands. See mailx(1) in the *User's Reference Manual.*

Directories in `/var`

This section describes the directories of the `/var` directory, which contain files and directories that vary from machine to machine.

`/var/adm`

This directory contains system logging and accounting files.

`/var/cron`

This directory contains the `cron` log file.

`/var/lp`

This directory contains log files for the LP print service.

`/var/mail`

This directory contains subdirectories and mail files that users access with the `mail`(1) and `mailx`(1) commands.

`/var/mail/:saved`

This directory contains temporary storage for mail messages while `mail` is running. Files are named with the user's ID while they are in `/var/mail`.

/var/news

This directory contains news files. The file names are descriptive of the contents of the files; they are analogous to headlines. When a user reads the news, using the news command, an empty file named .news_time is created in his or her login directory. The date (time) of this file is used by the news command to determine if a user has read the latest news file(s).

/var/opt

This directory is created and used by application packages.

/var/options

This directory contains a file (or symbolic link to a file) that identifies each utility installed on the system. This directory also contains information created and used by application packages (such as temporary files and logs).

/var/preserve

This directory contains backup files for vi and ex.

/var/sadm

This directory contains logging and accounting files for the backup and restore services, software installation utilities, and package management facilities.

/var/sadm/pkg

This directory contains data directories for installed software packages.

/var/saf

This directory contains log files for the Service Access Facility.

/var/spool

This directory contains temporary spool files.

/var/spool/cron/crontabs

This directory contains `crontab` files for the `adm`, `root`, and `sys` logins. Users whose login IDs are in the `/etc/cron.d/cron.allow` file can establish their own `crontab` file using the `crontab` command. If the `cron.allow` file does not exist, the `/etc/cron.d/cron.deny` file is checked to determine if the user should be denied the use of the `crontab` command.

As `root`, you can use the `crontab` command to make the desired entries. Revisions to the file take effect at the next reboot. The file entries support the `calendar` reminder service and the Basic Networking Utilities. Remember, you can use the `cron` function to decrease the number of tasks you perform with the `sysadm` command; include recurring and habitual tasks in your `crontab` file. (See `crontab(1)` in the *User's Reference Manual* for additional information.)

/var/spool/lp

This directory contains temporary print job files.

/var/spool/smtpq

This directory contains Simple Mail Transfer Protocol (SMTP) directories and log files. Directories named *host* contain messages spooled to be sent to that host. Files named `LOG.`*n* contain the logs from the past seven days (Sunday's log is called `log.0`). The current day's log is simply `LOG`.

/var/spool/uucp

This directory contains files to be sent by uucp.

/var/spool/uucppublic

This directory contains files received by uucp.

/var/tmp

This directory contains temporary files.

/var/uucp

This directory contains logging and accounting files for uucp.

Files in `/var`

This section describes the files in the `/var` directories, which contain information that varies from machine to machine.

`/var/adm/spellhist`

If the Spell Utility is installed, this file contains a history of all words that the `spell` command fails to match. Periodically, this file should be reviewed for words that you can add to the dictionary. Clear the `spellhist` file after reviewing it. (Refer to `spell(1)` in the *User's Reference Manual* for information on adding words to the dictionary, cleaning up the `spellhist` file, and other commands that can be used with the Spell Utility.)

`/var/adm/utmp`

This file contains information on the current system state. This information is accessed with the `who` command.

`/var/adm/utmpx`

This file contains information similar to that in the `/var/adm/utmp` file, along with a record of the remote host.

`/var/adm/wtmp`

This file contains a history of system logins. The owner and group of this file must be `adm`, and the access permissions must be 664. Each time `login` is run this file is updated. As the system is accessed, this file increases in size. Periodically, this file should be cleared or truncated. The command line `>/var/adm/wtmp` when executed by `root` creates the file with nothing in it. The following command lines limit the size of the `/var/adm/wtmp` file to the last 3600 characters in the file:

```
# tail -3600c /var/adm/wtmp > /var/tmp/wtmp
# mv /var/tmp/wtmp /var/adm/wtmp
#
```

The /usr/sbin/cron, /usr/sbin/rc0, or /usr/sbin/rc2 command can be used to clean up the wtmp file. You can add the appropriate command lines to the /var/spool/cron/crontabs/root file or add shell command lines to directories such as /etc/rc2.d, /etc/rc3.d, and so on.

/var/adm/wtmpx

This file contains information similar to that in the /var/adm/wtmp file, along with a record of the remote host.

/var/adm/loginlog

If this file exists, it is a text file that contains one entry for each group of five consecutive unsuccessful attempts to log in to the system.

/var/adm/sulog

This file contains a history of substitute user (su) command usage. As a security measure, this file should not be readable by others. The /var/adm/sulog file should be truncated periodically to keep the size of the file within a reasonable limit. The /usr/sbin/cron, the /usr/sbin/rc0, or the /usr/sbin/rc2 command can be used to clean up the sulog file. You can add the appropriate command lines to the /var/spool/cron/crontabs/root file or add shell command lines to directories such as /etc/rc2.d, /etc/rc3.d, and so on. The following command lines limit the size of the log file to the last 100 lines in the file:

```
# tail -100 /var/adm/sulog > /var/tmp/sulog
# mv /var/tmp/sulog /var/adm/sulog
#
```

/var/cron/log

This file contains a history of all actions taken by /usr/sbin/cron. The /var/cron/log file should be truncated periodically to keep the size of the file within a reasonable limit. The /usr/sbin/cron, /usr/sbin/rc0, or /usr/sbin/rc2 command can be used to clean up the /var/cron/log file. You can add the appropriate command lines to the /var/spool/cron/crontabs/root file or add shell command lines in the following directories (as applicable): /etc/rc2.d, /etc/rc3.d, (and so on). The following command lines limit the size of the log file to the last 100 lines in the file:

```
# tail -100 /var/cron/log > /var/tmp/log
# mv /var/tmp/log /var/cron/log
#
```

/var/sadm/bkup/logs/bklog

This file contains a process log used when troubleshooting a backup operation.

/var/sadm/bkup/logs/bkrs

This file contains a process log used when troubleshooting a backup or restore operation for which a method was not specified.

/var/sadm/bkup/logs/rslog

This file contains a process log used when troubleshooting a restore operation.

/var/sadm/bkup/toc

This file contains table of contents entries created by a backup method.

C Using the sysadm Interface

Introduction

This appendix explains how to use the system administration menu interface (accessed through the sysadm command) for UNIX System V Release 4.0. The menu interface is displayed differently on different terminal screens due to terminal specific characteristics such as window border characters and function key screen displays. The examples shown in this appendix are based on output generated while working on an AT&T 5620 terminal. (The sysadm menu interface supports all terminals supported by the Forms and Menus Language Interface.) For a summary of commands that allow you to navigate through the menu system on your terminal, see "Quick Reference to the sysadm Interface" in Chapter 1, "Overview of System Administration."

This appendix is organized as follows:

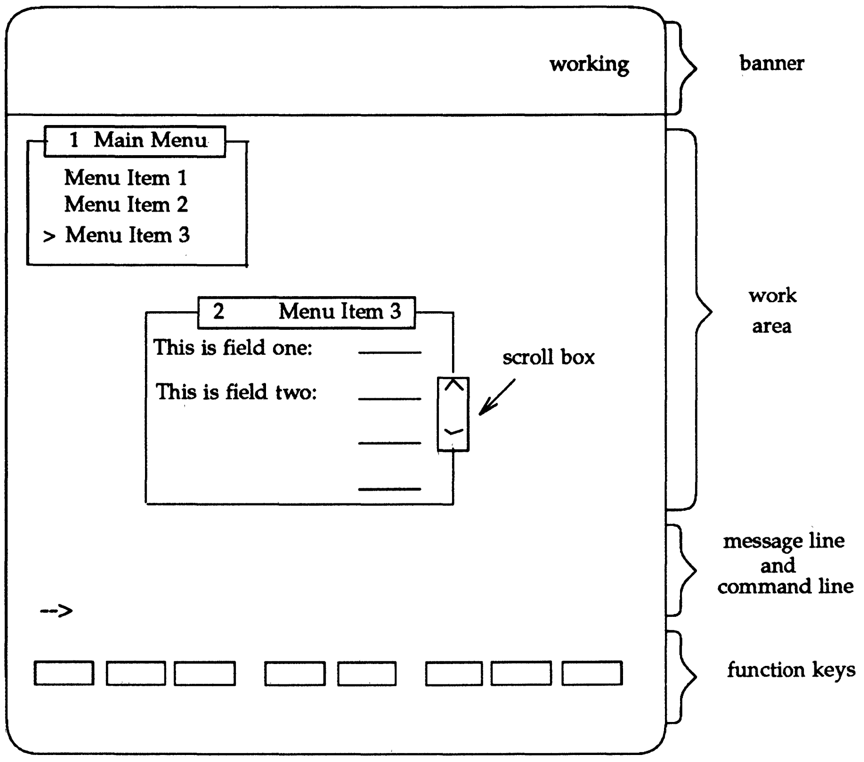
- "A Tour of the Menu Interface Window"
  This section introduces the components of a window as they appear during interactive sessions of menu use. (The term "window" is used instead of "screen" because you may have more than one window on your screen at a given time, but the menu interface requires only one window.)
- "Frame Manipulation Tools"
  This section explains how to move around the window while using the menus. Available tools include navigation keys (such as the up arrow key), function keys, alternate keystrokes (for users whose keyboards do not have function keys), and a menu of commands for use within the interface.
- "A Sample Session: Adding an Account for a New User"
  This section is a walk-through of a task that all system administrators must perform: adding an account for a new user to a system. The section shows exactly what the administrator does, using the menus, to accomplish this task. Both user input and computer output are shown in illustrations of a window as they would appear at various times during this procedure.
- Summary of Interface Procedures
  This section explains, in more detail, the procedures described in the sample session (see "A Sample Session" above), and documents several options not covered in the sample case. Specifically, this section explains how to do the following: enter and exit the interface, navigate among the windows, use menus and forms, and get help along the way.

- "System Administration Menus"
  This section provides a complete list of the menus and tasks available, through the sysadm command, in UNIX System V Release 4.0.

A Tour of the Menu Interface Window

The system administration menu interface provided with pre-Release 4.0 UNIX systems worked as a series of menus and prompts that scrolled off the top of the window as an administrator progressed through a task. The menu interface for UNIX System V Release 4.0, however, does not scroll off the window. Figure C-1 shows the organization of a window during a typical session with the menus.

Figure C-1: The System Administration Menu Interface Window



The window is divided into five areas: a banner line, a work area, a message line, a command line, and the function keys line. Figure C-1 also shows the icons used in the interface frames: the cursor (a right angle bracket); a prompt for special sysadm commands (an arrow), which appears only when invoked with a CTRL-J sequence; and the pointers to additional information not visible on a frame (up and down icons). The rest of this section defines all these components.

Window Areas

The five areas of the window are as follows:

banner line	The banner line is an area across the top of the window reserved for the word `working`, which appears to let you know that tasks are in progress and that new tasks cannot be initiated.
work area	The work area is used by the interface for presenting frames. A frame is an independently scrollable region of the window, enclosed in a "box." There are three basic types of frames: menus, tasks, and messages. The `sysadm` interface displays multiple frames simultaneously; as you invoke successive frames, they are arranged, in an overlapping format, on the window. The active frame is distinguished from other frames by its highlighted title.
	Each frame that displays one or more lines of information displays a scroll box in its right-hand border. Figure C-1 shows a scroll box containing both an "up icon" (`^`) and a "down icon" (`v`). When present, the up and down icons signal that there is more information than can be seen in the current frame, either before or after (respectively) the current frame. For example, the up indicator appears when a user is viewing page two of a multiple-page form.
message line	The message line is the place in the window where the menu interface displays one-line messages. These messages include brief instructions and error messages.

command line	The command line is available for interface commands that can be typed in addition to being selected from the command menu. It is marked by an arrow prompt (-->).
function keys	The display of eight boxes at the bottom of the window lists the function keys available for the current frame. Function keys are numbered keys that have been programmed to perform specific jobs, thus providing a convenient way of doing tasks you do frequently (such as obtaining a help message). Many keyboards have eight function keys (some have none and some have more than eight) and they are usually located in a row across the top of the keyboard. (If your keyboard does not have function keys, you can use a set of equivalent keystrokes that allow you to accomplish the same tasks performed by the keys. For details, see "Alternate Keystrokes" under "Frame Manipulation Tools" later in this appendix.) The sysadm menu interface displays a different subset of the function keys for each type of frame used by the interface. Thus, if you are working in a multi-page task form, eight function keys will be available, but if you are working in a menu, only six function keys will be available. The names of the function keys (which describe what they do) appear in the last line of the window. For definitions of the function keys available with the sysadm interface, see "Function Keys" under "Frame Manipulation Tools" later in this appendix.

Frame Icons

The following icons are used in a sysadm window:

Right angle bracket (>)	The right angle bracket serves as a cursor: it appears in the left-hand margin of a menu and points to an item on that menu. It is used for selecting menu items. To access a menu item, position the cursor on it and press the RETURN key (or the ENTER function key).
Down icon (v)	A "down icon " in the right-hand margin of a menu shows that more menu items exist below those visible in the frame. To access those items, press the "down icon" key on the right-hand side of your keyboard. The cursor will descend as the menu scrolls up, one item at a time.
Up icon (^)	An "up icon" in the right-hand margin of a menu shows that more menu items exist above those visible in the frame. To access those items, press the "up arrow" key on the right-hand side of your keyboard. The cursor will ascend as the menu scrolls down, one item at a time.
Right arrow (-->)	A right arrow appears as a special command prompt at the bottom of the window if you invoke it with a CTRL-J sequence. The special commands you can enter after it include `cancel`, `cleanup`, `exit`, `help`, `refresh`, and `update`, which are also available on the `Command Menu`. To invoke the special command prompt, enter a CTRL-J sequence.

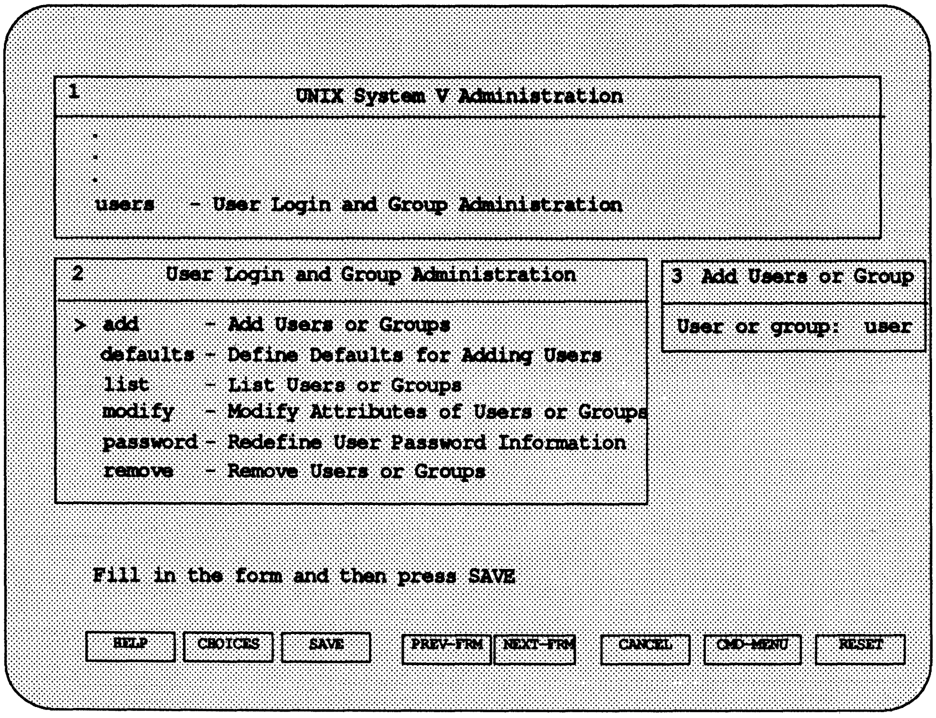
NOTE

When both the up icon and down icon are shown in the right-hand margin of a menu, more menu items exist above and below those visible in the frame. Use the up and down arrow keys to move the cursor on a menu.

Types of Frames

The frames displayed in the sysadm menu interface contain three types of information: menus, forms, and messages. Figure C-2 shows examples of two of these types in one window. The window in this figure is taken from a session in which an administrator is adding an account for a new user to the system.

Figure C-2: Adding an Account for a New User



```
1                    UNIX System V Administration
 .
 .
 .
 users    - User Login and Group Administration

2        User Login and Group Administration        3 Add Users or Group
> add      - Add Users or Groups                     User or group:  user
  defaults - Define Defaults for Adding Users
  list     - List Users or Groups
  modify   - Modify Attributes of Users or Groups
  password - Redefine User Password Information
  remove   - Remove Users or Groups

 Fill in the form and then press SAVE
```

In the work area of this window you can see two types of frames: menus (the main menu and `User Login and Group Administration` menu), and a task form (`Add Users or Group`). All three types of frames are discussed below.

Menus

A menu is a frame containing a list of other menus and tasks. In the example in Figure C-2, the menu shown in the upper left-hand corner is the main menu (the menu that appears when you type `sysadm`). Below it is the `User Login and Group Administration` menu, which has been selected from the main menu. This menu lists six tasks.

Tasks

A task, in the context of the menu interface, consists of a frame containing one or more prompts for information, each followed by a line on which a default response appears (when a default response is available). If you want to enter a response other than that shown on the line (or if no default is shown), invoke a menu of choices by pressing the CHOICES key, and select the appropriate item.

In Figure C-2, the frame labeled `Add Users or Group` is a task form for adding a user. This form appeared in the window when the task `add` was selected from the `User Login and Group Administration` menu.

Messages

The menu interface also provides help messages. Help messages provide background information designed to help you accomplish a particular task; they do not prompt you to provide information. You can get a help message by pressing the HELP function key; the message will appear in a frame.

Frame Manipulation Tools

To help you navigate among frames and respond to prompts within frames, the `sysadm` interface defines four tools: navigation keys, function keys, alternate keystrokes, and a set of commands for controlling the menu interface window. Each of these tools is described below.

Navigation Keys

NOTE **Not all keyboards have the keys described in this section.**

The following keys are available for navigating around a `sysadm` window. They are located on the left-hand or right-hand side of your keyboard.

Arrow Keys	Use the up and down arrow keys to scroll backward and forward, respectively, through a menu.
"Top" and "Bottom" Keys	These keys, sometimes provided on the left-hand side of your keyboard, allow you to skip to the first and last items, respectively, on a menu.
HOME	This key returns you to the first line of a menu.

Function Keys

Function keys are numbered, programmable keys to which the menu interface has assigned names and functions.

NOTE **If you are using a software application package written by a manufacturer other than AT&T, you may see function key definitions other than those described here, displayed on your screen. Refer to the documentation delivered with your software for definitions of these function keys.**

Function keys usually consist of a row of eight keys, placed across the top of the keyboard. (If your keyboard does not have function keys, see "Alternate Keystrokes" later in this section.) Each key is labeled fn where *n* is a number between 1 and 8. The keys are numbered consecutively from left to right.

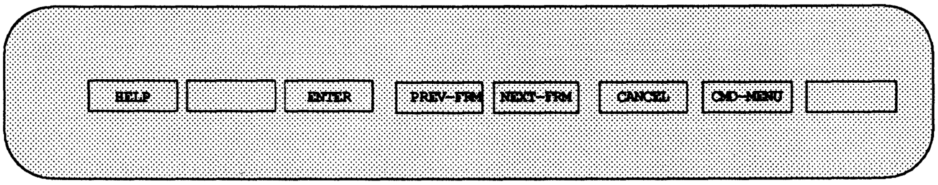
At the bottom of the window you'll see a row of eight boxes, lined up in parallel with the function keys on the keyboard. Each box contains the name of the function for the corresponding key, such as CANCEL or SAVE.

The sysadm interface defines thirteen functions keys. Not all of them are available during all sessions; the subset of available keys is determined by the type of activity you are doing.

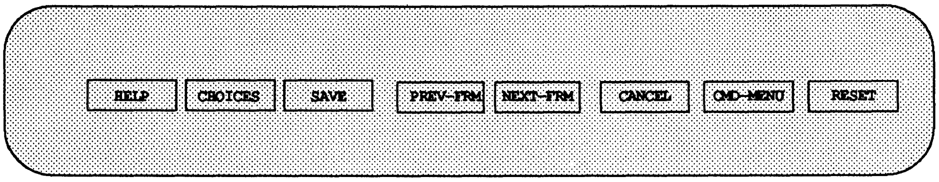
`CANCEL`	Deletes the current frame and returns you to the previous frame
`CHOICES`	Displays possible answers to a prompt
`CMD-MENU`	Displays a menu with special purpose commands for controlling the `sysadm` window
`CONT`	Resumes a task that was interrupted by a confirmation message–a query from the system asking you to verify that you want to continue the current task. (The system sends a confirmation message when you are doing a task that may have undesirable results to make sure you are aware of the risks at hand.)
`ENTER`	Selects the current menu item (also done with the RETURN key)
`HELP`	Displays text that describes what actions you may take at the current cursor position
`MARK`	Lets you select multiple items within a `CHOICES` menu
`NEXTFRM`	Moves you to the next frame
`NEXTPAGE`	Pages the text up, displaying the next page of information
`PREVFRM`	Moves you to the previous frame.
`PREVPAGE`	Pages the text down, displaying the previous page of information
`RESET`	Restores the default value to the current prompt
`SAVE`	Saves the answers currently displayed, and either displays a subsequent form (if there is one) or performs the task

The `sysadm` interface uses several types of menus, forms and messages: menus that list available menus and forms, choices menus, and the command menu; single-page and multi-page task forms; and help and confirmation messages. For each of these seven types, a different subset of the function keys is available. The following list shows which function keys are available for each type of frame:

The following function keys are available with menus.

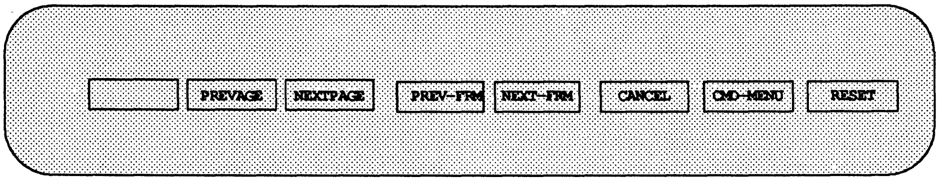


The following function keys are available with single-page task forms.

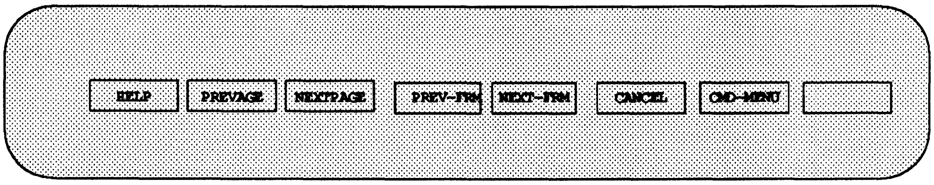


(The `RESET` key is not available with all single-page task forms.)

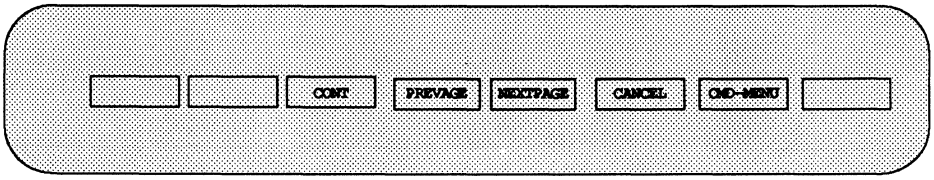
The following function keys are available with multi-page task forms.



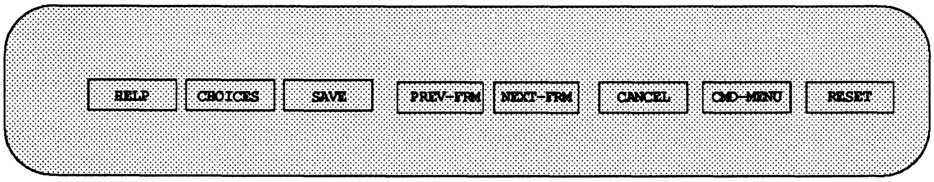
The following function keys are available with help messages.



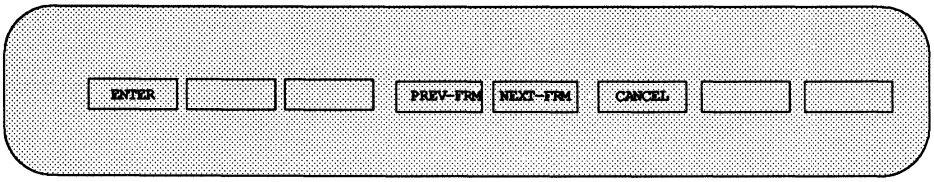
The following function keys are available with confirmation messages.



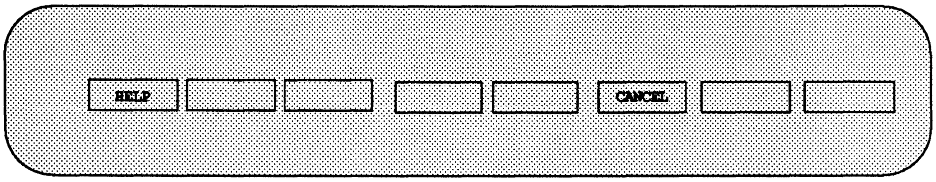
The following function keys are available with some choices menus.



The following function keys are available with other choices menus.



The following function keys are available with command menus.



As these lists show, there are two sets of keys that may be displayed with a choices menu; which one is displayed is determined by the type of task being done.

Alternate Keystrokes

If function keys are not provided or cannot be used with your keyboard, you can perform the functions done by these keys (such as saving a form and canceling a menu choice) by entering special character sequences instead. This type of sequence is called an "alternate keystroke." Each alternate keystroke consists of two components, entered in the order shown:

- CTRL-f (entered by holding down the key marked CTRL or CONTROL while you press "f")
- the number corresponding to the function key that does the desired task

For example, suppose you want to invoke a help message. You know that you can do so by pressing the f1 function key but you don't have function keys on your keyboard. Enter CTRL-f and then type 1. (Do not try to enter an alternate keystroke by pressing three keys–the CTRL key, the f key, and the *number* key–at once. Instead, enter the CTRL-f sequence first, and then press the appropriate numbered key.) The help message will appear on your screen.

Automatic Function Key Downloading

The `sysadm` interface was built with the FMLI (Forms and Menus Language Interface) tool. FMLI relies heavily on the use of a terminal's keyboard function keys F1 through F8 but these keys are not available on some terminals or are not designed in such a way that the System V `terminfo`(4) database can work with them cleanly. To help in these situations, FMLI provides equivalent two-character sequences to achieve the same effect: CTRL-f 1 through CTRL-f 8. Sometimes, however, new users of an FMLI application such as sysadm do not know about these equivalent character sequences, and are thus unable to use the screen-labeled function keys of the interface, a rather frustrating situation for such new users. FMLI 4.0 provides a partial fix to this problem of new users, at a possible slight inconvenience to some experienced users, on some terminals. If a terminal, as defined in its terminfo entry, does not have default escape sequences for its function keys but can download strings into its function keys (this is the case, for example, on the AT&T 5620 and 630 terminals), then FMLI attempts to download the equivalent two-character sequences into the function keys. This occurs, by default, when fmli is first invoked (in this case by sysadm) and each time that the user returns from any full-screen activity, such as one initiated with the run command. For terminals not fitting this description, no downloading is done.

For new users this default behavior is helpful, but for an experienced user who has already placed other strings into the function keys, this is inconvenient, as the FMLI character sequences replace the user's own, and FMLI cannot restore the user's automatically.

Using the LOADPFK Environment Variable to Disable Downloading

The environment variable LOADPFK can be used to specify that this downloading not be done. Setting LOADPFK=NO in the environment before invoking fmli (and thus sysadm) prevents this downloading from occurring at any time.

Using a Shell Script to Restore Your Function Key Settings

Users who have entered strings in their function keys on this type of terminal may find a shell script to restore their stings useful; the following script uses the tput utility to download function key strings:

```
tput pfx 1 'string-for-function-key-1'
tput pfx 2 'string-for-function-key-2'
tput pfx 3 'string-for-function-key-3'
tput pfx 4 'string-for-function-key-4'
tput pfx 5 'string-for-function-key-5'
tput pfx 6 'string-for-function-key-6'
tput pfx 7 'string-for-function-key-7'
tput pfx 8 'string-for-function-key-8'
```

Downloading FMLI Sequences More Efficiently

Even when a user wants the FMLI sequences downloaded, it is more efficient to do it only once if no program run from the FMLI application changes the function key settings (most programs do not); any possible delays required to download the sequences occur just once this way, instead of at the beginning of the application and every time a full-screen activity is run. This can be done using `tput`, as shown above, but with the FMLI two-character sequences. It is more easily done, however, by invoking `sysadm` in the following command sequence:

```
sysadm              # then exit immediately
LOADPFK=NO   sysadm
```

The first invocation of `sysadm` downloads the equivalent two-character sequences. You then exit from that invocation and call it again, with `LOADPFK=NO`, to have it run a little more efficiently.

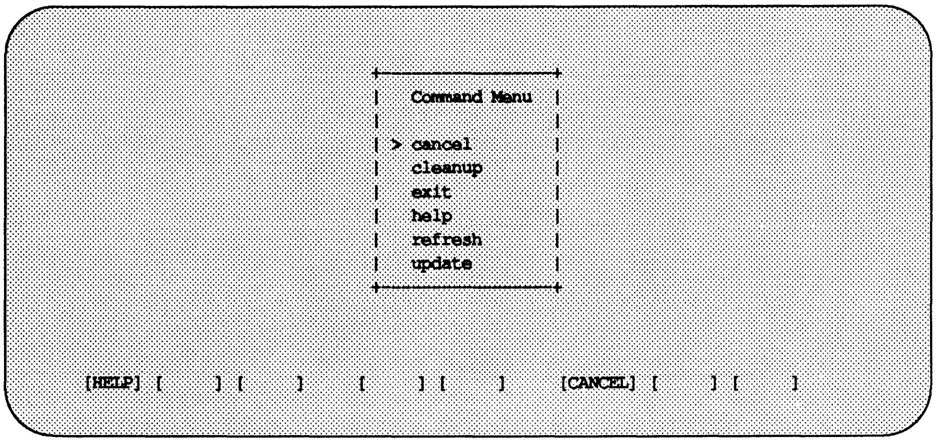
Frame Manipulation Commands

The `sysadm` interface provides a set of commands for navigating among frames and using the information in them. These commands may be entered by selecting them from a command menu or by typing them on the command line. You can use the commands at any time during your session with the `sysadm` interface. This section describes this set of frame manipulation commands.

The Command Menu

The `Command Menu` can be accessed by pressing the CMD-MENU function key. The following frame will appear:

Figure C-3: The Command Menu



```
                    +--------------------+
                    |   Command Menu     |
                    |                    |
                    | > cancel           |
                    |   cleanup          |
                    |   exit             |
                    |   help             |
                    |   refresh          |
                    |   update           |
                    +--------------------+

[HELP] [      ] [      ]   [      ] [      ]   [CANCEL] [      ] [      ]
```

Each command is defined below:

`cancel`	Cancels the last choice you made for the last menu displayed.
`cleanup`	Cancels everything in the menu interface window except the current menu and those menus that can be canceled only when you exit the menu interface.
`exit`	Leaves the `sysadm` menu interface and returns you to the shell prompt. After you issue this command, your screen will be cleared of everything except the shell prompt for root (`#`).
`help`	Provides information about a frame or a field in a frame, depending on your position in the window when you invoked help.
`refresh`	Refreshes (that is, clears and redisplays) the screen. This function is useful when the window becomes cluttered as a result of a console message.
`update`	Restores all default values to a form.

As Figure C-3 shows, two function keys are available when you are using the command menu: HELP and CANCEL.

`HELP`	Describes what actions you may take at the current cursor position
`CANCEL`	Deletes the current frame and returns you to the previous frame

The Command Line

As an alternative to invoking the `Command Menu`, the `sysadm` interface allows you to type the menu commands on a command line within the interface window. This command line is not always visible in your window: you must invoke it.

To invoke a command line prompt, enter a CTRL-] or CTRL-f c sequence: hold down the CTRL key while you press the] (or f) key. When you enter the control character, the interface displays an arrow prompt (`-->`) on the command line near the bottom of the window. (Both control character sequences are described in "Automatic Function Key Downloading" above.)

You can navigate among frames by entering the CTRL-] or CTRL-f sequence and then typing the frame number (the number in the upper left-hand corner of each frame) and pressing RETURN.

A Sample Session: Adding an Account for a New User

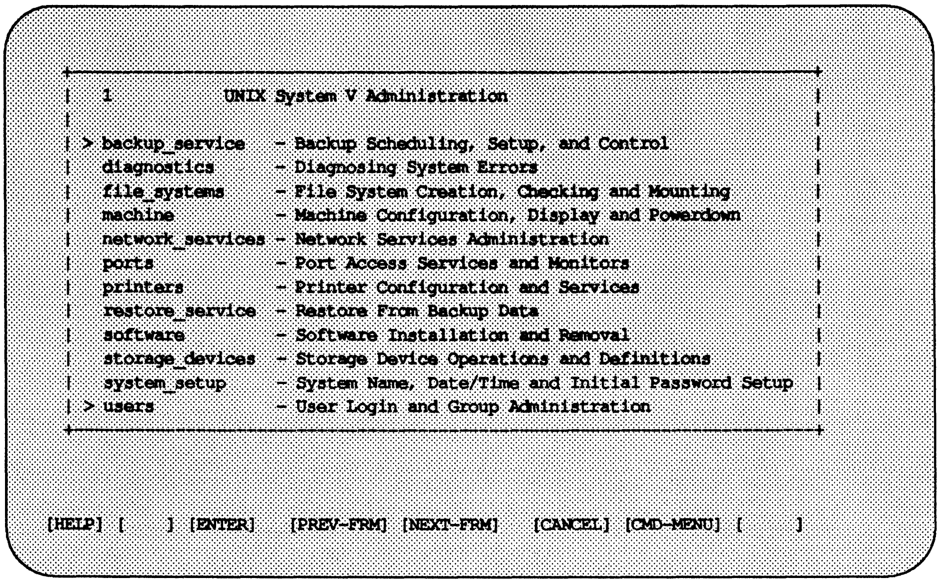
In this section we will show you exactly what you must do to add an account for a new user to your system. Assume that you are working on an AT&T 5620 terminal.

Step 1: Access the Menus

The first step is to access the interface. To do this, type `sysadm`. You will be prompted to enter your terminal type and password. Then the system administration main menu will appear in a frame in your terminal window, as shown in Figure C-4.

NOTE The type of terminal will affect the layout of the menus on your screen; more menu items will be immediately visible on some types of terminals rather than on others. The types of software you have installed will affect the entries you see on the main menu. For example, if you have installed one or more application packages on your system, your main menu will include an entry called `Applications`. This entry will not appear on your main menu if you have no application packages installed.

Figure C-4: System Administration Main Menu



```
+--------------------------------------------------------------------------+
|  1          UNIX System V Administration                                 |
|                                                                          |
| > backup_service   - Backup Scheduling, Setup, and Control               |
|   diagnostics      - Diagnosing System Errors                            |
|   file_systems     - File System Creation, Checking and Mounting         |
|   machine          - Machine Configuration, Display and Powerdown        |
|   network_services - Network Services Administration                     |
|   ports            - Port Access Services and Monitors                   |
|   printers         - Printer Configuration and Services                  |
|   restore_service  - Restore From Backup Data                            |
|   software         - Software Installation and Removal                   |
|   storage_devices  - Storage Device Operations and Definitions           |
|   system_setup     - System Name, Date/Time and Initial Password Setup   |
| > users            - User Login and Group Administration                 |
+--------------------------------------------------------------------------+


[HELP] [    ] [ENTER]  [PREV-FRM] [NEXT-FRM]   [CANCEL] [CMD-MENU] [     ]
```

All the entries on the main menu represent other menus. The entries on these second-level menus (and the menus that can be accessed through them), however, include both menus and tasks.

Step 2: Select a Menu

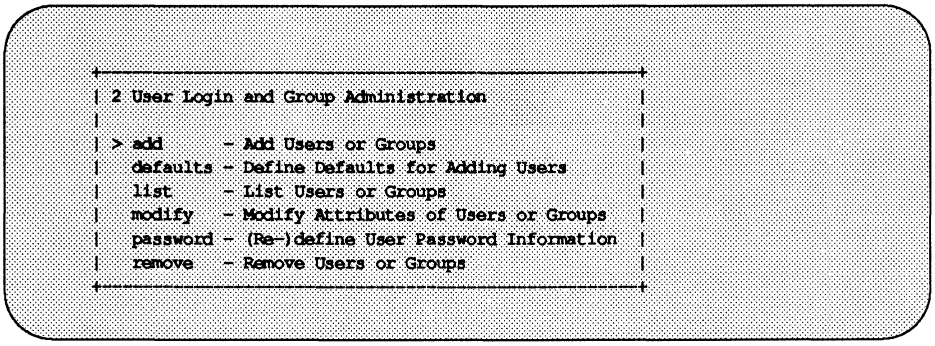
Because you want to add a login account for a new user, you need the `User Login and Group Administration` menu (abbreviated as `users`). There are two ways you can move the cursor to this item.

- Use the down arrow key to position the cursor on the `users` menu item. Then press RETURN.
- Enter the first letter or letters of the `users` menu entry: `u`. Then press RETURN.

  Typing `u` is sufficient because there is only one menu entry that begins with the letter u. When more than one menu entry has the same initial letter, you must request each one by entering the first two letters of its name. (If the initial two letters are the same for multiple entries, you must enter the first three letters of the desired menu name; if the initial three letters are the same, you must enter four; and so on.)

Figure C-5 shows how the system responds to either type of request: by displaying the menu you have specified.

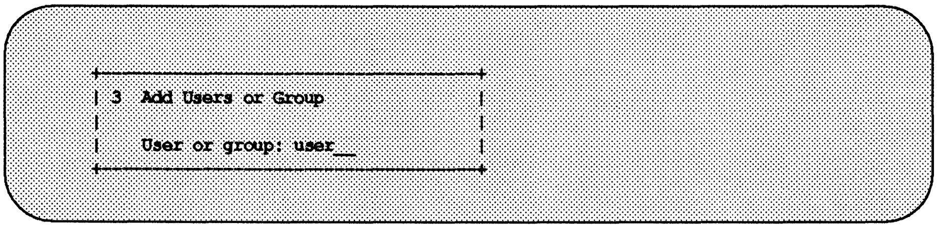
Figure C-5: Step 2: The Menu Selected from the Main Menu



```
+---------------------------------------------------------+
| 2 User Login and Group Administration                   |
|                                                         |
| > add      - Add Users or Groups                        |
|   defaults - Define Defaults for Adding Users           |
|   list     - List Users or Groups                       |
|   modify   - Modify Attributes of Users or Groups       |
|   password - (Re-)define User Password Information      |
|   remove   - Remove Users or Groups                     |
+---------------------------------------------------------+
```

Step 3: Select a Task

Now select a task (from the `User Login and Group Administration` menu) that will let you add an account for a new user to your system. The `add` task (short for `Add Users or Groups`) allows you to do this. Select this task the same way you selected a menu item in Step 2: by either positioning the cursor on the desired task (and pressing RETURN) or by typing the first one or two letters of the task you want (and pressing RETURN). Most often, once you have selected a task, the first thing you will see is a task form in a new frame. This is what happens when you select the `add` task.



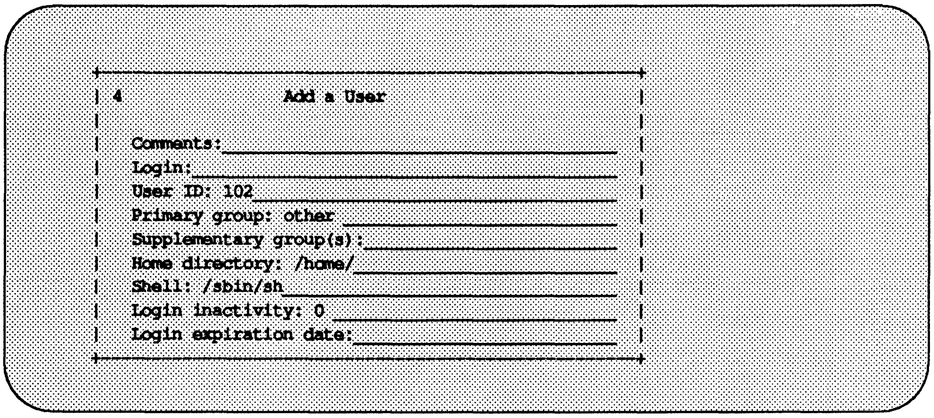
```
+-------------------------------------+
| 3  Add Users or Group               |
|                                     |
|    User or group: user__            |
+-------------------------------------+
```

Step 4: Fill Out the Forms

The frame displayed when you select a task is called a task form. A task form is an electronic questionnaire; you're expected to fill it out.

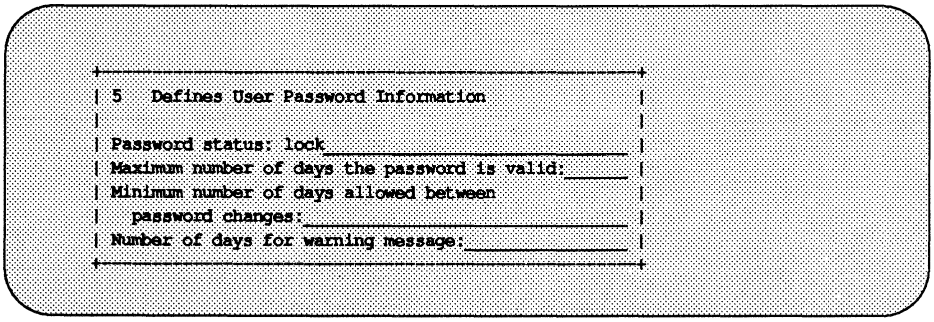
The task form displayed in your window when you select `add` contains an "either/or" type of question, for which only two answers are possible. The `user` choice appears after the prompt because it is the default selection. (Default values are not available for all task forms.) Because we want to add a user to the system, the default is the correct choice. Press the SAVE function key (or enter the alternate keystroke CTRL-f 3) so that this value will be accepted as input for your task.

The interface responds by presenting a detailed form on which you are expected to enter information about the new user being added to the system. For most forms there are default values for at least some of the information requested, as shown in the following example.



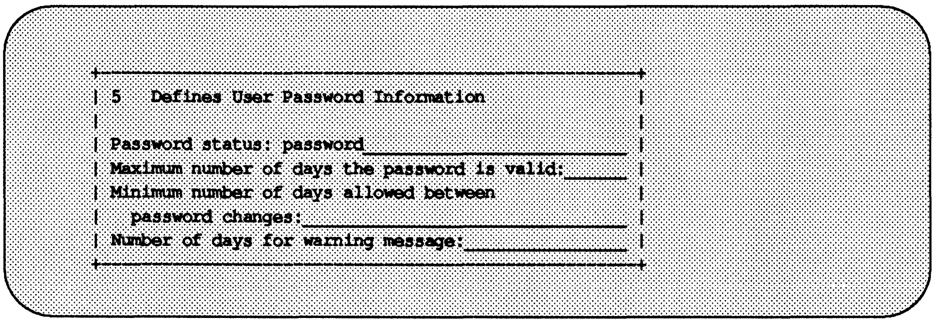
```
+--------------------------------------------------------+
| 4                  Add a User                          |
|                                                        |
|   Comments:___________________________________________ |
|   Login:______________________________________________ |
|   User ID: 102________________________________________ |
|   Primary group: other _______________________________ |
|   Supplementary group(s):_____________________________ |
|   Home directory: /home/______________________________ |
|   Shell: /sbin/sh_____________________________________ |
|   Login inactivity: 0 ________________________________ |
|   Login expiration date:______________________________ |
+--------------------------------------------------------+
```

Fill out the form and then press SAVE to enter the information on your form. Provide values for the following three mandatory fields: `Comments`, `Login`, and `Login expiration date`. Filling out the rest of the fields is optional. The system will accept this information and then present a third form, prompting you for information that can be used in assigning a password to the new user.



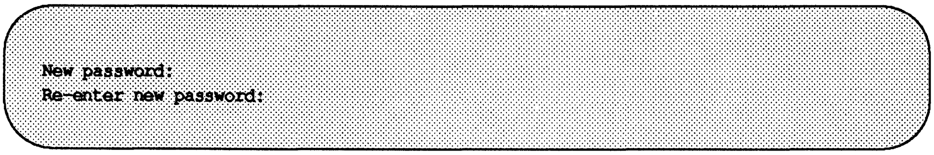
```
+--------------------------------------------------------+
| 5  Defines User Password Information                   |
|                                                        |
| Password status: lock__________________________________|
| Maximum number of days the password is valid:__________|
| Minimum number of days allowed between                 |
|   password changes:____________________________________|
| Number of days for warning message:____________________|
+--------------------------------------------------------+
```

Invoke the `CHOICES` menu (by pressing the `CHOICES` key) and select `password` to assign a temporary password for the new user. The word `password` will replace the word `lock` on the `Password status:` prompt line.



```
+--------------------------------------------------------------+
| 5   Defines User Password Information                        |
|                                                              |
| Password status: password________________________________    |
| Maximum number of days the password is valid:_______         |
| Minimum number of days allowed between                       |
|   password changes:______________________________________    |
| Number of days for warning message:______________________    |
+--------------------------------------------------------------+
```

When you have finished filling out the password information form, the task changes to full window mode and the following prompts appear:



```
New password:
Re-enter new password:
```

Enter a password after the prompt and press RETURN. The password you assign will be valid only until the first time the new user to whom it's assigned logs in. At that time, the user will be prompted to select another password.

After you re-enter the temporary password, the system will confirm the password information. Press the CONT key and you will be returned to the `Add a User` form. At this point you can add another user or CANCEL out of the task.

NOTE For details about how to move from frame to frame, see "Navigating Among Frames" later in this chapter. For more information about manipulating a single frame, see "Frame Manipulation Commands" earlier in this chapter.

Step 5: Exit from the Interface

To exit from the menus and return to the shell, invoke the `Command Menu` by pressing the CMD-MENU function key or by entering a CTRL-f 7 sequence. Select the `exit` command by positioning the cursor on it (with the arrow keys) and pressing RETURN. The screen will be cleared of all the frames, text, and icons produced by the menu interface, and a shell prompt will appear.

Summary of Interface Procedures

This section explains how to do the following:

- Navigate among frames
- Use function keys
- Use tasks and fill out forms
- Get help along the way–from help messages and the `CHOICES` menus
- Use the express mode from the UNIX system shell command line, so you can perform `sysadm` functions without navigating through menus.

Navigating Among Frames

You can move back and forth among frames by using one of three function keys: PREV-FRM, NEXT-FRM, and CANCEL. The function keys are the keys on your keyboard, labeled f1 through f8, for which the menu interface has defined specific operations.

NOTE If your keyboard does not have function keys, you can substitute the following keystroke sequence: 1) enter CTRL-f (pronounced "control-f") by holding down the CTRL key while you press the f key; 2) then press the key with the same number as the function key you want to access. For example, to access the function represented by the f1 key, enter the CTRL-f sequence and then press 1.

The names of the operative function keys are displayed at the bottom of the window. Not all function keys are available for all types of frames; see "Frame Manipulation Tools" earlier in this appendix for a list of function keys available with the various types of frames and for definitions of all the function keys.

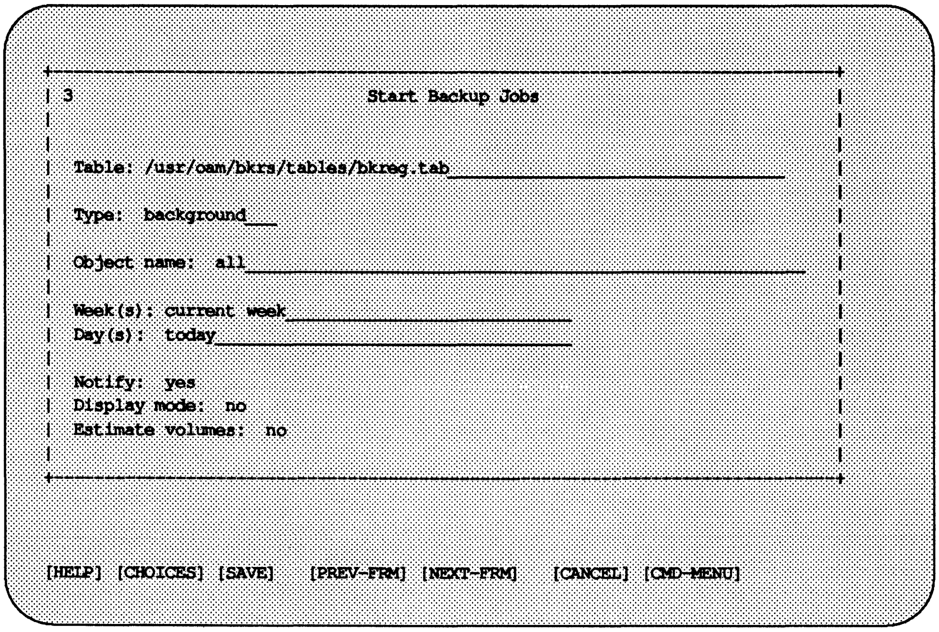
Using Tasks and Forms

All menu entries are one of two types: other menus or tasks. Functionally, a task is an administrative activity, such as adding an account for a new user or backing up a file system. Within the sysadm menu interface, a task can be one of three types: a form-based task, a display task, or a full-window task. Each task can be identified as one of these types as soon as you select it. The rest of this section describes each type of task in detail.

Form-Based Tasks

A form-based task is an activity that the interface cannot do until it has received certain information from you. (Most of the tasks in the sysadm menu interface are form-based tasks.) To solicit this information, the task displays one or more forms on the screen. A form is an electronic questionnaire: a list of questions followed by blank spaces (in which you can provide answers) or default answers. Figure C-6 shows an example of a form:

Figure C-6: A Sample Form: Start Backup Jobs



```
+------------------------------------------------------------------------+
| 3                          Start Backup Jobs                           |
|                                                                        |
|                                                                        |
| Table: /usr/oam/bkrs/tables/bkreg.tab__________________               |
|                                                                        |
| Type:  background___                                                   |
|                                                                        |
| Object name:  all________________________________                     |
|                                                                        |
| Week(s): current week________________                                  |
| Day(s):  today_______________________                                  |
|                                                                        |
| Notify:  yes                                                           |
| Display mode:  no                                                      |
| Estimate volumes:  no                                                  |
|                                                                        |
+------------------------------------------------------------------------+


[HELP] [CHOICES] [SAVE]   [PREV-FRM] [NEXT-FRM]   [CANCEL] [CMD-MENU]
```

Notice that there are answers after some of the prompts. These answers are default values provided by the menu interface. If a default answer is acceptable, simply press RETURN after it. If it is not acceptable, type your own answer over it before pressing RETURN.

Notice, too, that seven function keys are available with a task form: HELP, CHOICES, SAVE, PREV-FRM, NEXT-FRM, CANCEL, and CMD-MENU.

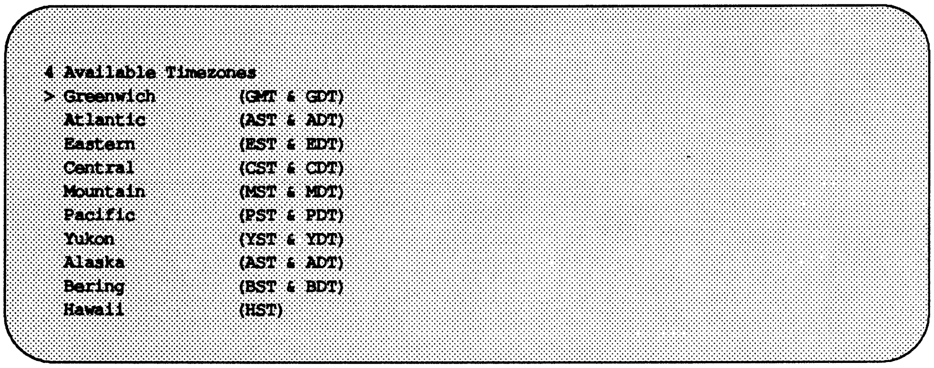
Filling Out a Task Form

The menu interface helps you fill out forms by giving you a choice of possible answers for some prompts. To access a list of valid choices, press the CHOICES function key. One of two things will happen.

If there are only two possible answers to the current prompt, the interface will display one answer after the prompt in the form. You can use that answer, or you can press the CHOICES key again, to have the second answer displayed after the prompt (overwriting the first answer). If you want to see both answers before selecting one, you can toggle back and forth between them by pressing the CHOICES key repeatedly.

If there are three or more possible answers to the current prompt, the interface will display a "pop-up menu" from which you can make a selection. Figure C-7 shows a sample pop-up menu: the menu of valid specifications for a time zone that is displayed when you are setting the time zone for your system.

Figure C-7: Sample Pop-Up Menu: Valid Time Zones



```
4 Available Timezones
> Greenwich         (GMT & GDT)
  Atlantic          (AST & ADT)
  Eastern           (EST & EDT)
  Central           (CST & CDT)
  Mountain          (MST & MDT)
  Pacific           (PST & PDT)
  Yukon             (YST & YDT)
  Alaska            (AST & ADT)
  Bering            (BST & BDT)
  Hawaii            (HST)
```

If your terminal is equipped with arrow keys (up, down, left, and right), you can use the up and down arrow keys to move the cursor to the menu item you want. You can press the TAB key to scroll through a form, but not to scroll through a menu.

After you have typed an answer to a prompt (or found a valid answer on a pop-up menu), you must press SAVE to have your choice accepted as input for the current task. If you want to change your answer after entering it, press CANCEL and make a new selection.

If you need more information about what to do at any particular time, press the HELP function key. A brief message about what to do (at the current cursor position) will be displayed.

Changing Answers on a Form

Even after you have answered all the questions on a form, you can change your answer to a particular question. If your terminal is equipped with arrow keys (up, down, left, and right), you can use these keys to move the cursor to the answer you want to change. When the cursor is pointing to the appropriate item, type your new answer over the original one.

If your terminal does not have arrow keys, press the TAB key to cycle through the questions. When you reach the appropriate item, type your new answer over the original one.

Having Your Answers Validated

The answer you supply will be validated for correctness. If it is invalid, an error message will appear on the message line. Select HELP or CHOICES for more information about valid answers.

Saving the Answers on a Form

Once you are satisfied with your answers on a form, press SAVE to tell the system that you are ready to proceed to the next step. If there are more forms to be filled out, the interface will display the next one. If there are no more forms, the system will perform the specified task and display a report about the results.

Example of Filling Out a Task Form

Say you want to remove, from your system, the account for a user named Diane White. Begin by requesting the `User Login and Group Administration` menu from the main `sysadm` menu. From that menu, in turn, select the `remove` task.

At this point, the interface will ask you whether you want to remove an account for a user or an account for a group, by displaying the following prompt:

```
User or group: user
```

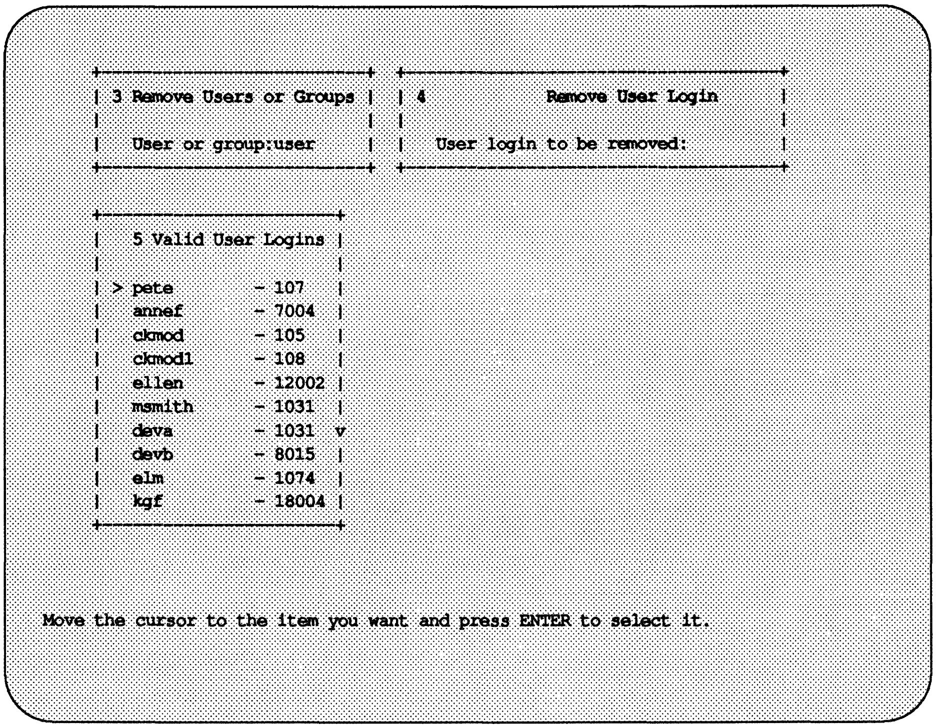
The word `user`, the default answer, already appears after the prompt. Because you want to remove an account for an individual user (named Ellen), this is the correct answer for you. Press the SAVE function key to have this information saved.

The interface will respond by opening a new frame containing another question:

```
User login to be removed:
```

There is no default answer to this question and you don't know Ellen's login. Therefore you press the CHOICES function key. A pop-up menu appears (as shown in Frame 5 of Figure C-8), displaying a list of all the valid logins for your computer.

Figure C-8: Example of Filling Out a Form



```
+------------------------+  +---------------------------------------+
| 3 Remove Users or Groups |  | 4             Remove User Login       |
|                          |  |                                       |
|   User or group:user     |  |   User login to be removed:           |
+------------------------+  +---------------------------------------+

+----------------------+
|   5 Valid User Logins |
|                       |
| > pete       - 107    |
|   annef      - 7004   |
|   ckmod      - 105    |
|   ckmod1     - 108    |
|   ellen      - 12002  |
|   msmith     - 1031   |
|   deva       - 1031   v
|   devb       - 8015   |
|   elm        - 1074   |
|   kgf        - 18004  |
+----------------------+

Move the cursor to the item you want and press ENTER to select it.
```

From this list, you can see that Ellen's login is `ellen`. Move the cursor to the `ellen` entry and press the RETURN key. The `CHOICES` menus disappears and the chosen login is printed in the `Remove User Login` form. Then press the SAVE function key.

A new form, called `Remove a User Login`, will appear in your current window. This form will contain some information about the login for the relevant user. Only one prompt, however, will require a response from you; you must answer the last question on the form, `Remove home directory and all files?`. Complete this form and press SAVE.

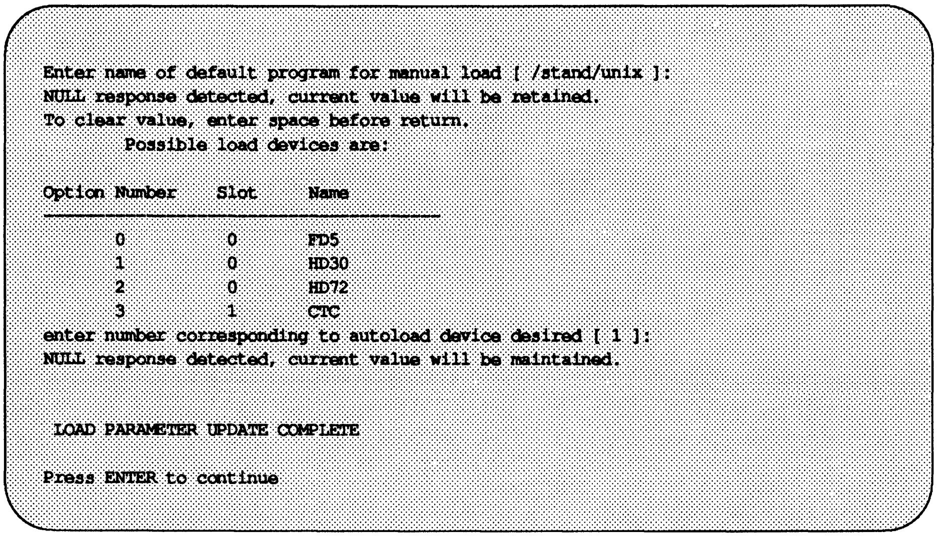
After you've filled out the form and saved it, a confirmation frame will appear to let you know that you have successfully removed the account for the owner of the login `ellen`. To remove the confirmation frame, press CANCEL.

Full Window Tasks

A full window task is an activity that involves an interactive function outside the menu interface. This function may consist of a series of forms (such as those used in form-based tasks), a prompt for information, or a piece of text to be read. Full window tasks may be any of these types; their classification together is based on the fact that when one of these tasks is invoked, the menu interface temporarily disappears (the current window is cleared of all text and graphics).

Figure C-9 shows a sample of a full window task.

Figure C-9: A Sample Full Window Task



```
Enter name of default program for manual load [ /stand/unix ]:
NULL response detected, current value will be retained.
To clear value, enter space before return.
        Possible load devices are:

Option Number    Slot     Name
________________________________________
      0           0       FD5
      1           0       HD30
      2           0       HD72
      3           1       CTC
enter number corresponding to autoload device desired [ 1 ]:
NULL response detected, current value will be maintained.


 LOAD PARAMETER UPDATE COMPLETE

Press ENTER to continue
```

Notice that none of the graphical symbols and text associated with the menu interface (such as function key names) appear in this window; the entire window is occupied by task related text.

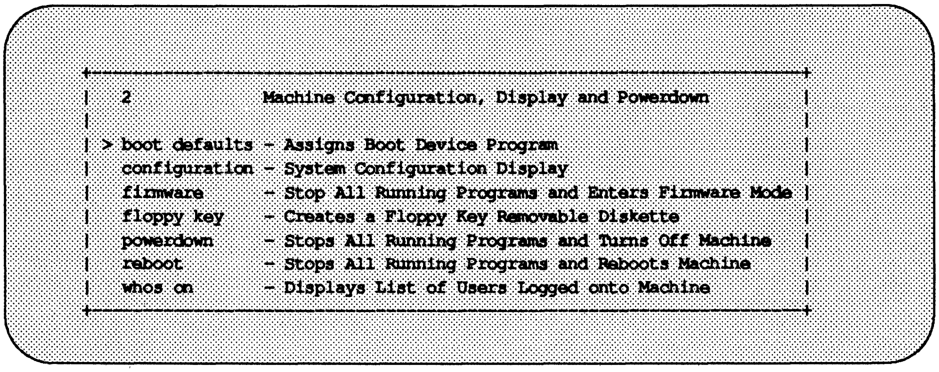
The only text from the menu interface that appears in all full window tasks is a prompt at the bottom of the window: `Press ENTER to continue.` If you do so, you will be returned to the `sysadm` menu interface. All text other than this line is specific to a particular task; other than having this line, full window tasks may all differ from one another.

When you press ENTER or RETURN, you reenter the menu interface. Then you have access, once again, to the function keys, `Command Menu`, and other features of the menu interface.

Example of a Full Window Task

Assume you have invoked the `Machine Configuration, Display and Powerdown` menu, as shown in Figure C-10.

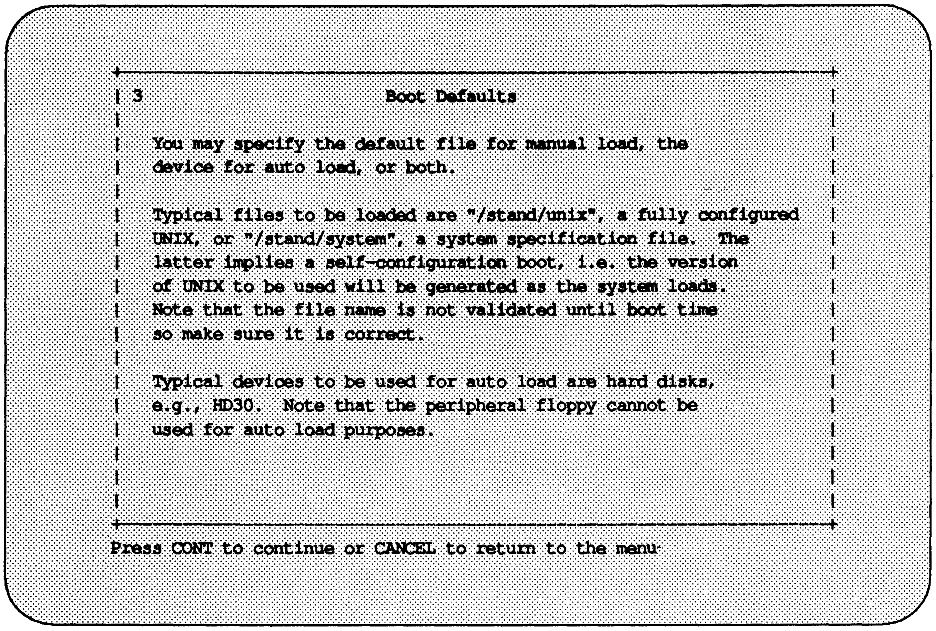
Figure C-10: The `sysadm machine` Menu



```
+-----------------------------------------------------------------------------+
|  2             Machine Configuration, Display and Powerdown                 |
|                                                                             |
| > boot defaults - Assigns Boot Device Program                               |
|   configuration - System Configuration Display                              |
|   firmware      - Stop All Running Programs and Enters Firmware Mode        |
|   floppy key    - Creates a Floppy Key Removable Diskette                   |
|   powerdown     - Stops All Running Programs and Turns Off Machine          |
|   reboot        - Stops All Running Programs and Reboots Machine            |
|   whos on       - Displays List of Users Logged onto Machine                |
+-----------------------------------------------------------------------------+
```

You select the `boot defaults` task and another frame appears, titled `Boot Defaults`. (This frame is not a full window.)

Figure C-11: Frame Produced by Selecting `Boot Defaults`



```
+------------------------------------------------------------------+
| 3                      Boot Defaults                             |
|                                                                  |
|  You may specify the default file for manual load, the          |
|  device for auto load, or both.                                  |
|                                                                  |
|  Typical files to be loaded are "/stand/unix", a fully configured|
|  UNIX, or "/stand/system", a system specification file.  The    |
|  latter implies a self-configuration boot, i.e. the version     |
|  of UNIX to be used will be generated as the system loads.      |
|  Note that the file name is not validated until boot time       |
|  so make sure it is correct.                                     |
|                                                                  |
|  Typical devices to be used for auto load are hard disks,       |
|  e.g., HD30.  Note that the peripheral floppy cannot be         |
|  used for auto load purposes.                                    |
|                                                                  |
|                                                                  |
|                                                                  |
+------------------------------------------------------------------+
Press CONT to continue or CANCEL to return to the menu
```

Because the next step in this task involves the display of a full window, you are then prompted to press CONT rather than RETURN to continue. When you press CONT, everything will be cleared from the current window and the following prompt will appear at the top: `Enter name of default program for manual load [ ]:`.

When you have finished reading all the text and filling out all the forms associated with this task, the prompt `Press ENTER to continue` will appear. Then you may return to the menu interface.

Getting Help

When you don't know what to do next, or what information you are expected to provide to the interface, press the HELP key; information about the frame in which the cursor is currently positioned will be displayed.

NOTE You cannot access help messages when you are working on a full-window task.

If you press the HELP key while a menu is being displayed, you will get a help message for the current menu item (which is highlighted). If you press the HELP key while a form is being displayed, you will get a help message for the current prompt.

Using Express Mode

Once you are familiar with the menu interface, you may want to take advantage of a menu or task without navigating through the menus. The `sysadm` interface provides a way to do this: express mode. Express mode allows you to access a menu or task directly from the UNIX system shell command line. To access a menu or task from the shell, type `sysadm`, followed by the name of the desired menu or task. For example, if you want to change the password for a user on your system, and you know that the task called `password` on the `Users` menu will allow you to do that, you can simply type `sysadm password` to access that task. Because the name of this task (`password`) is unique within the menu interface, the form for that task is displayed on the screen.

If, however, you specify a menu or form that does not have a unique name, the interface will display a menu of matching items, from which you can select the one you wanted.

If you know that the item you want to specify does not have a unique name, but you know its location in the menu system, you can request it by specifying its menu pathname. For example, to access the `add` task from the `User and`

`Group Management` menu (abbreviated as `users`), you can simply enter the following:

```
        sysadm users/add
```

If the task name you typed cannot be located, the system displays a message saying so.

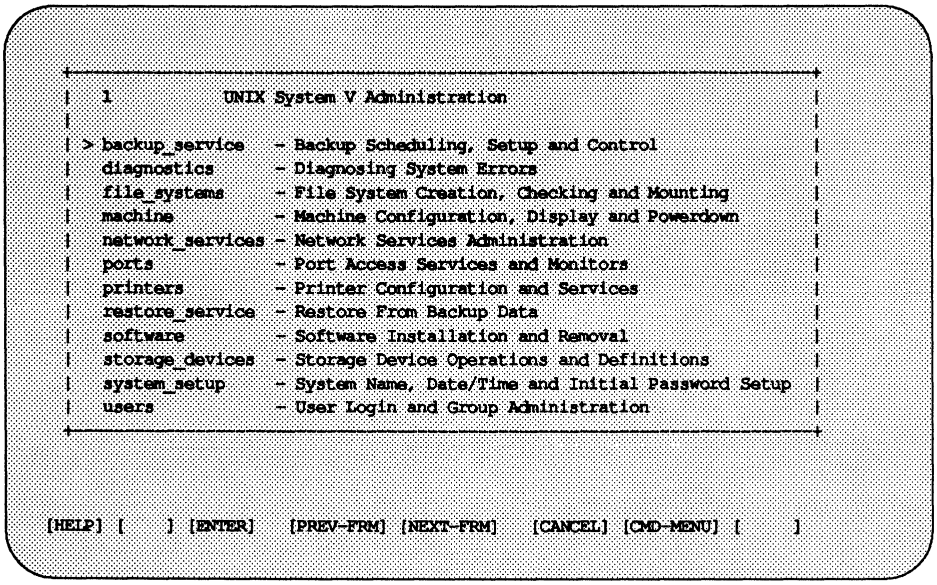
NOTE

Keep in mind, when you use express mode, that you will not be able to access any menus or task forms that appear on the main menu or on any menu higher in the menu tree than the menu you originally accessed. (For a chart showing the entire menu tree see "The System Administration Menus" at the end of this appendix.) For example, if you have typed `sysadm password` at the shell prompt, you will not be able to move to the `Users` menu (the menu directly above the `password` task in the menu tree) by pressing CANCEL (as you would be able to do had you accessed the menus without express mode).

System Administration Menus

Administrative tasks are grouped together under entries on the main menu, which is invoked by typing `sysadm`.

Figure C-12: System Administration Main Menu



```
+-------------------------------------------------------------------------------+
|  1            UNIX System V Administration                                    |
|                                                                               |
| > backup_service   - Backup Scheduling, Setup and Control                     |
|   diagnostics      - Diagnosing System Errors                                 |
|   file_systems     - File System Creation, Checking and Mounting              |
|   machine          - Machine Configuration, Display and Powerdown             |
|   network_services - Network Services Administration                          |
|   ports            - Port Access Services and Monitors                        |
|   printers         - Printer Configuration and Services                       |
|   restore_service  - Restore From Backup Data                                 |
|   software         - Software Installation and Removal                        |
|   storage_devices  - Storage Device Operations and Definitions                |
|   system_setup     - System Name, Date/Time and Initial Password Setup        |
|   users            - User Login and Group Administration                      |
+-------------------------------------------------------------------------------+



[HELP] [     ] [ENTER]   [PREV-FRM] [NEXT-FRM]   [CANCEL] [CMD-MENU] [      ]
```

The following table lists the menus on the system administration main menu and all the subsidiary menus and tasks available with each entry. Main menu entries appear in the left-hand column. The center column lists the menus and tasks available through each main menu entry. The right-hand column lists the tasks offered by each menu in the center column.

Main Menu Item	Menu Item or Task	Task
backup_service	backup	
	history	full
		limit
		selective
	reminder	add
		display
		modify
		remove
	respond	
	schedule	add
		display
		modify
		remove
	setup	add
		exception_list
		full
		modify
		remove
		rotation
		summary
	status	full
		limit
		modify
		selective
diagnostics	diskrepair	
	diskreport	

Main Menu Item	Menu Item or Task	Task
file systems	check	
	defaults	add
		display
		modify
		remove
	diskuse	
	display	
	fileage	
	filesize	
	identify	
	list	
	make	
	mount	
	umount	
machine	boot default	
	configuration	summary
		system
	firmware	
	floppy key	
	powerdown	
	reboot	
	whos on	
network_services	basic_networking	
	remote files	local resources
		remote resources
		setup
		specific ops
	selection	display
		modify
	name_to_address	loopback
		starlan
		npack

Main Menu Item	Menu Item or Task	Task
ports	port_monitors	add
		disable
		enable
		list
		modify
		remove
		start
		stop
	port_services	add
		disable
		enable
		list
		modify
		remove
	tty_settings	add
		list
printers	classes	add
		list
		modify
		remove
	filters	add
		list
		modify
		remove
		restore
	forms	add
		list
		modify
		remove

Main Menu Item	Menu Item or Task	Task
	operations	accept
		control
		disable
		enable
		mount
		reject
		set default
		unmount
	printers	add
		list
		modify
		remove
	priorities	list
		set default
		set priority
	requests	cancel
		hold
		move
		release
	status	forms
		printers
		requests
		wheels
	systems	add
		list
		modify
		remove
restore_service	operator	
	respond	
	restore	
	status	full
		modify
		selective

Main Menu Item	Menu Item or Task	Task
software	check	installed
		original
		spooled
	defaults	add
		list
		modify
		remove
	install	
	interact	
	list	
	read_in	
	remove	
storage_devices	copy	
	descriptions	add
		attributes
		list
		remove
		reservation
	display	
	erase	
	format	
	groups	add
		list
		membership
		remove
	partition	
	remove	
system_setup	datetime	display
		set
	nodename	display
		set
	password	
	setup	

Main Menu Item	Menu Item or Task	Task
users	add	
	default	
	list	
	modify	
	password	
	remove	

B. CUSTOMIZING THE sysadm INTERFACE

D Customizing the sysadm Interface

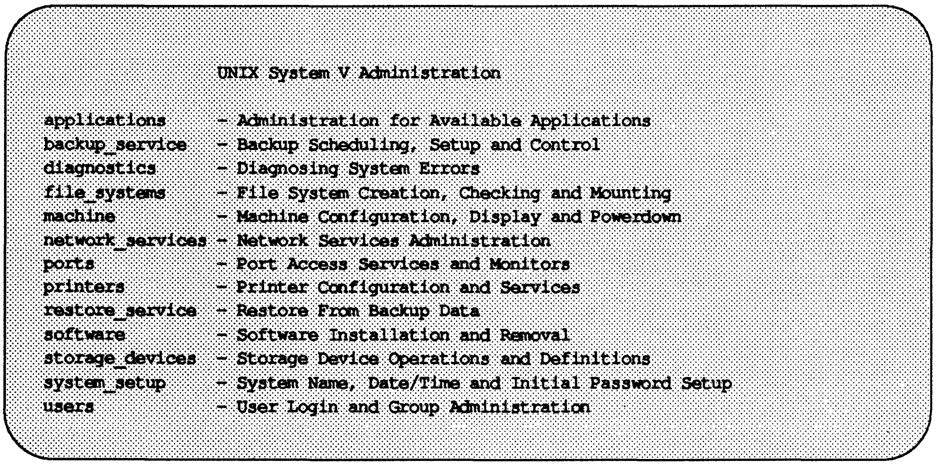
Overview of Customizing the sysadm Interface

UNIX System V Release 4 provides a menu interface to the most common administrative procedures. It is invoked by executing the sysadm command and so is referred to as the sysadm interface.

Additions or changes can be made directly to this interface by using two shell commands, edsysadm and delsysadm, or they can be delivered as part of a software package. This appendix provides instructions for making direct changes to the interface using edsysadm and delsysadm. Instructions for delivering changes in a package are covered in the *Programmer's Guide: System Services and Application Packaging Tools*.

The Interface Structure: a Hierarchy of Menus

The sysadm interface consists of a hierarchy of menus. (For a thorough description of this hierarchy, see the "Using the sysadm Interface" appendix.) At the top of the hierarchy is the main menu (labeled `UNIX System V Administration`). It appears on your screen, immediately after you invoke sysadm, as follows:



```
                    UNIX System V Administration

applications     - Administration for Available Applications
backup_service   - Backup Scheduling, Setup and Control
diagnostics      - Diagnosing System Errors
file_systems     - File System Creation, Checking and Mounting
machine          - Machine Configuration, Display and Powerdown
network_services - Network Services Administration
ports            - Port Access Services and Monitors
printers         - Printer Configuration and Services
restore_service  - Restore From Backup Data
software         - Software Installation and Removal
storage_devices  - Storage Device Operations and Definitions
system_setup     - System Name, Date/Time and Initial Password Setup
users            - User Login and Group Administration
```

NOTE

The applications menu will not appear on the main menu until at least one menu or task has been placed under it.

Menus and Tasks

The main menu consists of a list of function specific menus. The lefthand column notes the menu names (such as `machine`) and the righthand column gives descriptions of these menus. Each menu offers other menus and/or names of tasks. For example, the `machine` menu, shown below, contains one menu (`configuration`) and six tasks.

```
           Machine Configuration Display and Powerdown

boot defaults - Assigns Boot Device Program
configuration - System Configuration Display
firmware      - Stop All Running Programs and Enters Firmware Mode
floppy key    - Creates a Floppy Key Removable Diskette
powerdown     - Stops All Running Programs and Turns Off Machine
reboot        - Stops All Running Programs and Reboots Machine
whos on       - Displays List of Users Logged onto Machine
```

Choosing the menu entry from this screen will cause another menu to be presented. Choosing a task entry will begin execution of that task.

Selecting a Name and Location for an Interface Entry

Before making a change to the interface, (either by modifying an existing menu or task or by adding a new one) you will need to know the name, description, and location of the menu or task being added or changed, as defined below.

Name	The name of the menu or task as it will appear in the left-hand column of the screen.
Description	The description of the menu or task as it will appear in the righthand column of the screen.
Location	The location of a menu or task in the `sysadm` menu hierarchy. This location is a combination, step-by-step, of all the menu names that must be chosen to reach the menu or task. Each step must already exist when the entry is added. For example,when you add a task with a location of `main:applications:mypkg`, you must already have created an entry for the menu `mypkg`. All locations begin with `main`. When defining a location in the procedures that follow, each step should be separated by a colon. For example, the `powerdown` task is under the menu `machine`, which, in turn, is under the `main` menu. Thus, the location of the `powerdown` task is `main:machine`. Refer to the section entitled "System Administration Menus" in the "Using the `sysadm` Interface" appendix for a complete listing of the menus.

You can create new `sysadm` menus at any level and you can change or add to any of the original `sysadm` menus. You should be aware, however, that if you make changes to original menus you might cause problems in the execution of standard `sysadm` operations. It is therefore recommended (though not mandatory) that you create new menus for your interface changes. The main `sysadm` menu includes a menu entry called `applications` that is actually empty when your system is delivered. It is recommended (though not mandatory) that you use this menu to store any new menus and tasks that are associated with software application packages.

The edsysadm Command

The edsysadm command, which allows you to make changes or additions to the interface, is an interactive command that functions much like the sysadm command itself. It presents a series of prompts for information. (Which prompts appear depends on your response to them.)

After you have responded to all the prompts, edsysadm presents a form that you must fill in with information describing the menu or task being changed or added. This form is called the menu (or task) definition form. If you are changing an existing menu or task entry, the definition form will already be filled in with the current values, which you can edit. If you are adding a new menu or task entry, the form will be empty and you will have to fill it in.

This appendix describes the prompts presented by edsysadm, describes your choice of responses for each prompt, and tells you which response to choose for a particular course of action. Illustrations showing how these prompts appear on the screen are not included.

Item Help Files for Menus and Tasks

For every menu or task that you add with the edsysadm command, you must also provide an item help file. The messages in this file will be shown on a user's screen when that user requests help while on a menu screen or within a form. You can create an item help file with any editor but it must follow the format described in the "Writing Your Help Messages" section.

Because an item help file is required for every menu and task, you cannot skip the field for the name of the file when filling out the menu or task definition form. If you fill in this field with a filename that cannot be found, you are placed in an editor and can create an item help file at that time. edsysadm creates this new file in your current working directory and names it Help.

Executable Files for Tasks

Before adding a task to a menu, you must already have the executable files that will be run when a user chooses that task. You may write these yourself or a programmer may supply them to you. In either case, this appendix covers only the procedures used to make the changes to the interface. Instructions for writing the executables can be found in the *Programmer's Guide: System Services and Application Packaging Tools*.

The `delsysadm` Command

The `delsysadm` command allows you to delete a task or menu from the interface. It checks for dependencies on the entry being removed before it deletes it. (A dependency exists if the menu being removed contains an entry placed there by an application package.) If a dependency is found, you are asked whether you want to continue with the removal.

When you delete a menu entry, it must already be empty (contain no other menus or tasks). To remove a menu and all its entries at the same time, you can execute `delsysadm` with the `-r` option.

CAUTION Use `delsysadm` to remove only those menu or task entries that you have added to the interface with `edsysadm`.

Writing Your Help Messages

You must write help messages for every interface modification you make. They must be in what is called an item help file. This file has text for two types of messages:

- the help message that will be shown when the user requests help from the parent menu
- the help messages that will be shown for each field when your task action is a FACE (Framed Access Command Environment) form

The format of the item help file allows you to create one item help file for each task, combine all of your help messages for multiple tasks into one file, use the same message for multiple FACE forms, and to define a title hierarchy for the help message screens.

The Item Help File

There are no naming restrictions for the item help file when it resides on your machine. However, when it is placed into the interface structure, the item help file must always be named `Help`. Since `edsysadm` creates the directory structure required by the interface, it gives the file you name on the menu (or task) definition the appropriate location and correct name regardless of its name on your machine. This means that you do not have to name your item help files `Help` and, therefore, can have more than one item help file in your working directory at the same time. `edsysadm` will handle the details of giving it the correct name.

There are three types of entries in an item help file:

- the menu item help
- the default title (can define both a global default and a form default)
- the field item help

A description of each type of entry and its format follows. All of the entries use the colon (`:`) as the keyword delimiter.

The Menu Item Help Message Format

The menu item help will be shown whenever a user requests help while viewing the menu screen and the cursor is on the task or menu item. Menu item help must be written for both menu and task entries. For example, if you add a menu under `main:applications` and that menu has three tasks under it, you will need to deliver four menu item help messages.

The format for the menu item help definition is as follows:

> `[`*task_name*`:]ABSTRACT:`
> TAB *Line 1 of message text*
> TAB *Line 2 of message text*
> TAB *Line n of message text*

task_name defines the task (or menu) entry to which this help message belongs. This name must match the name that you have decided should appear in the lefthand column of the menu screen. (Refer back to "Selecting a Name and Location for an Interface Entry" for more details on this name.) *task_name* is not optional when more than one menu item help definition is defined in the same item help file. This helps in identifying the task or menu to which the message belongs.

The message text should be entered beneath the header line. There can be multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab character. Blank lines may be included within the message as long as they also begin with a tab character. An example menu item help definition is shown below.

```
task1:ABSTRACT:
      This is line one of the menu item help message.
      This is a second line of message text.

      The preceding line will appear as a blank line
      when the help message is shown because it begins
      with a tab.
```

The title for a menu item help message is always the description text, as it appears in the lefthand column of the menu display, prepended by the string `Help on`.

The Default Title Format

You can define two types of default titles:

- a global default title to be used on all of the help messages defined in the item help file
- a form default title to be used on all of the help messages defined for a particular form in an item help file with messages defined for numerous FACE forms (delivered as task actions)

Defaults can be overridden, as described in the section named "The Title Hierarchy." A default title definition is recommended but not required.

The format for the default title definition is as follows:

[*form_id*:]`TITLE`:*Title Text*

form_id is the name of the form as it is defined with `lininfo` in your FACE form definition. When a *form_id* is supplied, this line defines a form default title. When it is not supplied, this line defines a global default title.

The title text defined after the `TITLE` keyword will have the string `HELP on` prepended to it when displayed. Keep this in mind when writing the title.

An example form default title definition is shown below.

```
task1:TITLE:Package Administration Task1
```

If `task1` had not been added before `TITLE`, this example would be defining a global default title. The title defined by the example above will be displayed as:

```
HELP on Package Administration Task1
```

The Field Item Help Message Format

The field item help message will be shown whenever a user requests help from within a FACE form. Each field on the form must have a help message defined in the item help file.

The format for the field item help definition is as follows:

> [*form_id*:]`field_id`: [*Title Text*]
> TAB *Line 1 of message text*
> TAB *Line 2 of message text*
> TAB *Line n of message text*

form_id is the name of the form as it is defined with `lininfo` in your FACE form definition. When one item help file contains messages for multiple tasks (and so multiple forms), it is used to distinguish with which form a field belongs. It is optional if the file contains messages for only one task. `field_id` is the name of the field as it is defined with `lininfo` in your FACE form definition. *Title text* defines a title used only with the help message for this field. As with the default title, the text defined here will have the string `HELP on` prepended to it when displayed.

The message text should be entered beneath the header line. There can be multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab character. Blank lines may be included within the message as long as they also begin with a tab character. An example field item help definition is shown below.

```
task1:fld1:the Name Field
      This is the text for a field item help for a name
      field.

      The preceding line will appear as a blank line
      when the help message is shown because it begins
      with a tab.
```

The title for this field item help message, as defined above, will be `HELP on the Name Field.`

The Title Hierarchy

You can define a global default title, a form default title, and a field title in an item help file. When all three are defined in the same file, the following rules are followed:

- The global default title is used for any message defined in an item help file that does not have a form default title or field title.
- The form default title is used for any message defined in an item help file and that is associated with the form, unless it has a field title.
- The field title is used only for the one help message for which it is defined.

In summary, if no field title is defined, the form default title is used. If no form default title is defined, the global default title is used. You always want at least a global default title defined; otherwise, the string `HELP` on will be displayed with no descriptive text.

To define a global default title, add a line to your item help file in the following format:

> `TITLE`:*Title Text*

where *Title Text* is the text for the global default title.

To define a form default title, add a line to your item help file in the following format:

> *form_id*:`TITLE`:*Title Text*

where *form_id* is the name of form as it is defined with `lininfo` in your FACE form definition and *Title Text* is the text for the form default title.

To define a field title, use the following format for the field item help header line:

> *form_id*:*field_id*:*Title Text*

where *form_id* is the name of the form as it is defined with `lininfo` in your FACE form definition, *field_id* is the name of the field as it is defined with `lininfo` in your FACE form definition and *Title Text* is the text for the field title.

NOTE In all cases, the text defined as *Title Text* is always prepended with the string `HELP on` when displayed to a user.

Setting Up for Item Help in a FACE Object

To help the interface read your item help file and know with which forms and fields a help message is associated, the `help` and `lininfo` descriptors in the FACE object definition must be defined as follows:

- The `help` descriptor must be defined exactly as shown on the line below:

  ```
  help=OPEN TEXT $INTFBASE/Text.itemhelp $LININFO
  ```

- The `lininfo` descriptor for each field must be defined as

  `lininfo=`[*form_id*:]*field_id*

  where *form_id* and *field_id* are names each no longer than 30 characters. The names defined here as *form_id* and *field_id* must match exactly those used as *form_id* and *field_id* in the item help file.

NOTE Since no FACE form definition is created for a menu entry, no setup actions are required. However, you should be certain that the *task_name* keyword precedes the `ABSTRACT` heading line for a menu entry help message.

Example Item Help Files

This section shows two example item help files. Figure D-1 shows an item help file that defines messages for only one FACE form. Figure D-2 shows an example of defining messages for multiple FACE forms in one item help file.

Figure D-1: Item Help File for One Form

```
ABSTRACT:
         The text defined here will be shown to
         users when they request help while
         viewing the parent menu for this
         task.  The task name is "adding users."

TITLE:Adding Users

field1:
         The text defined here will be shown to
         users when they request help from the
         form and the cursor is positioned at
         field1.  The title for this message will
         be ``HELP on Adding Users'' as defined above.

field2:Field 2
         The text defined here will be shown to
         users when they request help from the
         form and the cursor is positioned at
         field2.  The title for this message will
         be ``HELP on Field 2''.
```

Note: The lininfo descriptors in the form definition associated with this file should look like this:

```
    .
    .
    .

lininfo=field1

    .
    .
    .

lininfo=field2
```

Figure D-2: Item Help File for Multiple Forms

```
add:ABSTRACT:
          The text defined here will be shown to
          users when they request help while
          viewing the parent menu for the task
          named add.

add_user:TITLE:Adding Users

add_user:field1:
          The text defined here will be shown to
          users when they request help from the
          form and the cursor is positioned at
          field1.  The title for this message will
          be ``HELP on Adding Users'' as defined above.

add_user:field2:Field 2
          The text defined here will be shown to
          users when they request help from the
          form and the cursor is positioned at
          field2.  The title for this message will
          be ``HELP on Field 2''.

delete:ABSTRACT:
          The text defined here will be shown to
          users when they request help while
          viewing the parent menu for the task
          named delete.

delete_user:TITLE:Deleting Users

delete_user:field1:
          The text defined here will be shown to
          users when they request help from the
          form and the cursor is positioned at
          field1.  The title for this message will
          be ``HELP on Deleting Users'' as defined above.

delete_user:field2:Field 2
          The text defined here will be shown to
          users when they request help from the
          form and the cursor is positioned at
          field2.  The title for this message will
          be ``HELP on Field 2''.
```

(continued on next page)

Figure D-2: Item Help File for Multiple Forms (continued)

Note: The lininfo descriptors in the form definition associated with this file should look like this:

```
    .
    .
lininfo=add_user:field1
    .
    .
lininfo=add_user:field2
    .
    .
lininfo=delete_user:field1
    .
    .
lininfo=delete_user:field2
```

Creating or Changing a Menu Entry

The procedures for creating a new menu and for changing an existing one are similar and both result in the display of a menu definition form. Each procedure is described below, followed by a description of the menu definition form.

Creating a Menu Entry

Before creating a new menu entry, you should:

- Select a name and description for the menu
- Select a location for it in the interface
- Prepare an item help file for the menu entry (refer to the "Writing Your Help Messages" section earlier in this chapter for instructions)

1. Type `edsysadm` and press RETURN.

   NOTE If you do not execute this command from the directory in which the item help file resides, supply the full pathname when prompted for the name of the item help file.

2. You are asked to choose between a menu and a task. Choose `menu` and press RETURN.
3. You are asked to choose between adding a new menu and changing an existing one. Choose `add` and press RETURN.
4. You are given an empty menu definition form. Fill it in and press SAVE. (See "The Menu Definition Form" for descriptions of the fields on this form.)
5. You are asked if you want to test the changes before actually making them. Answer `yes` or `no` and press SAVE. (If you answer yes, refer to the "Testing Your Menu Changes On-Line" section to learn what the test involves.)
6. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose `install` and press SAVE.

7. You will see one of the following:
   - A confirmation screen if the installation was successful. This means you now have an empty menu. (To populate the menu entry you just created, follow the procedure for adding a new menu entry or for adding a new task entry.)
   - An error message describing the reason installation was unsuccessful.
   - A message explaining that there is a conflict with the name you have chosen for this menu. You will then be prompted to choose from three possible actions: `install` (the addition will be installed despite the collision of names), `rename` or `relocate` (you will be shown the menu form at which time you can change the name or location of the entry), and do not `install`.

Changing a Menu Entry

Before changing a menu entry, you should:

- Know the name and description of the menu entry
- Know its location in the interface
- Change the associated item help file, if necessary, or create a new one (see the "Writing Your Help Messages" section earlier in this chapter for instructions)

1. Type `edsysadm` and press RETURN.

NOTE If you have changed an item help file or created a new one and you do not execute this command from the directory in which the file resides, supply the full pathname when asked for the name.

2. You are asked to choose between a menu and a task. Choose menu and press RETURN.

3. You are asked to choose between adding a new menu and changing an existing one. Choose `change` and press RETURN.
4. You are asked if your change is for an on-line menu or for a menu that has been saved for a package. Choose `on-line` and press SAVE.
5. You are asked to supply the name and location of the menu to be changed. Fill in the blanks and press SAVE.
6. You are given a menu definition form filled in with the current values for the menu named above. Make the desired changes and press SAVE. (See "The Menu Definition Form" for descriptions of the fields on this form.)
7. You are asked if you want to test the changes before actually making them. Answer `yes` or no and press SAVE. (If you answer yes, refer to the section entitled "Testing Your Menu Changes On-Line" to learn what the test involves.)
8. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose `install` and press SAVE.
9. You will see one of the following:
   - A confirmation screen if the installation was successful. This means that the menu has been changed.
   - An error message describing the reason installation was unsuccessful.
   - A message explaining that there is a conflict with the name you have chosen for this menu. You will then be prompted to choose from three possible actions: `install` (the changes will be installed despite the collision of names), `rename` or `relocate` (you will be shown the menu form at which time you can change the name or location of the entry), and `do not install`.

Testing Your Menu Changes On-Line

Before installing your menu changes, you may want to verify that you've added an entry to a menu. The `edsysadm` command gives you a chance to do this after you fill in the menu definition form. Follow these steps to perform your test.

1. Type `yes` when `edsysadm` presents the following prompt:

   ```
   Do you want to test this modification before continuing?
   ```

2. The parent menu (on which your addition or change is listed) is displayed. Check to make sure your modification has been made correctly.
3. Put the cursor on the new or changed menu entry and press the HELP key. The text of the help message for that menu entry is displayed so you can check it. (Press CANCEL to return to the menu.)
4. To exit on-line testing, press the CANCEL key.
5. You are returned to the prompt:

   ```
   Do you want to test this modification before continuing?
   ```

   If you want to continue executing the change, type no.

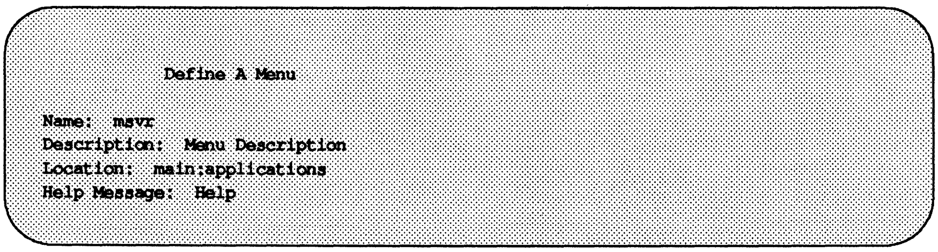
   If you want to make additional modifications to the menu definition form, press CANCEL. You are returned to the form and can make further changes at that time. (Press SAVE when you have finished your editing. You can then retest your changes or continue executing the change.)

The Menu Definition Form

This form contains four fields in which you must provide the following information: a menu name, a menu description, a menu location, and the name of the help message for the menu. Below are descriptions of the information you must provide in each field.

Menu Name	The name of the new menu (as it should appear in the lefthand column of the screen). This field has a maximum length of 16 characters and should consist of alphanumeric characters.
Menu Description	A description of the new menu (as it should appear in the righthand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except an at sign (`@`), circumflex (`^`), tilde (`~`), back quote (`` ` ``), single quote (`'`), and double quotes (`"`).
Menu Location	The location of the menu in the menu hierarchy, expressed as a menu pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this menu. Each menu name must be separated by colons. For example, the menu location for a menu entry being added to the `Applications` menu is `main:applications`. Do not include the menu name in this location definition. The complete path to this menu entry will be the menu location plus the menu name defined at the first prompt. This field has a maximum length of 50 characters and should consist of alphanumeric characters.
Menu Help File Name	Pathname to the item help file for this menu entry. If it resides in the directory from which you invoked `edsysadm`, you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by `$EDITOR`) to create one. The new file is created in the current directory and named `Help`.

The following screen shows a sample menu definition form that has been filled in:



```
            Define A Menu

Name:  msvr
Description:  Menu Description
Location:  main:applications
Help Message:  Help
```

Creating or Changing a Task Entry

The procedures for creating a new task and for changing an existing one are similar and both result in the display of a task definition form. Each procedure is described below, followed by a description of the task definition form.

Creating a Task Entry

Before creating a task entry, you should:

- Gather all files that will be associated with this task, such as the help files, FACE forms, and other executables. All files should already be prepared. Instructions for creating these files are provided in the *Programmer's Guide: System Services and Application Packaging Tools*.
- Decide on the task name and description
- Decide on its location in the interface
- Create an item help file (see "Writing Your Help Messages" earlier in this chapter for instructions)

1. Type `edsysadm` and press RETURN.

   NOTE: If you do not execute this command from the same directory in which the files associated with this task reside, enter full pathnames when supplying filenames.

2. You are asked to choose between a menu and a task. Choose `task` and press RETURN.
3. You are asked to choose between adding a new task or changing an existing one. Choose `add` and press RETURN.
4. You are given an empty task definition form. Fill it in and press SAVE. (See "The Task Definition Form" for descriptions of the fields on this form. Be aware that, when you name the menu under which you want this new task to reside, that menu must already exist.)
5. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose `install` and press SAVE.

6. You will see one of the following:
   - A confirmation screen if the installation was successful. This means that the new task is now in the menu and can be executed.
   - An error message describing the reason installation was unsuccessful.
   - A message explaining that there is a conflict with the name you have chosen for this task. You will then be prompted to choose from three possible actions: `install` (the addition will be installed despite the collision of names), `rename` or `relocate` (you will be shown the task form at which time you can change the name or location of the entry), and `do not install`.

Changing a Task Entry

Before changing a task entry, you should:

- Gather any of the files associated with this task that have been changed or are new. All files should already be prepared or changed. Instructions for creating these files are provided in the *Programmer's Guide: System Services and Application Packaging Tools*.
- Know the menu name and description
- Know its location in the interface
- Change the associated item help file, if necessary (see "Writing Your Help Messages" earlier in this chapter for instructions)

1. Type `edsysadm` and press RETURN.

NOTE If your change requires new files or changes to existing files and you do not execute this command from the directory in which the files reside, enter full pathnames when supplying filenames.

2. You are asked to choose between a menu and a task. Choose `task` and press RETURN.
3. You are asked to choose between adding a new task and changing an existing one. Choose `change` and press RETURN.
4. You are asked if your change is for an on-line task or for a task that has been saved for a package. Choose `on-line` and press SAVE.
5. You are asked to supply the name and location of the task to be changed. Fill in the blanks and press SAVE.
6. You are given a task definition form filled in with the current values for the task named above. Make the desired changes and press SAVE. (See "The Task Definition Form" for descriptions of the fields on this form.)
7. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose `install` and press SAVE.
8. You will see one of the following:
   - A confirmation screen if the installation was successful. This means that the change is in place.
   - An error message describing the reason installation was unsuccessful.
   - A message explaining that there is a conflict with the name you have chosen for this task. You will then be prompted to choose from three possible actions: `install` (the changes will be installed despite the collision of names), `rename` or `relocate` (you will be shown the task form at which time you can change the name or location of the entry), and `do not install`.

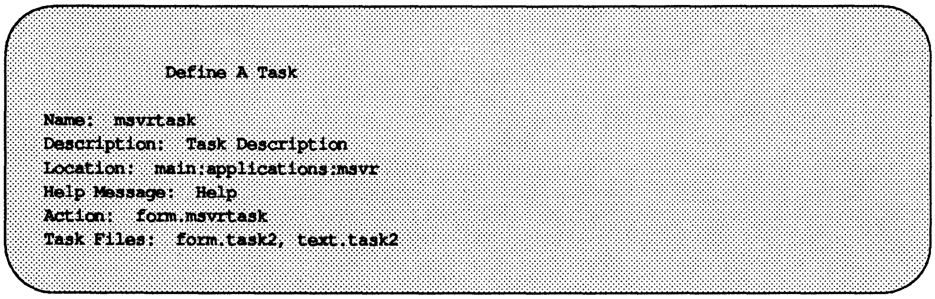
The Task Definition Form

This form contains six fields in which you must provide the following information: a task name, a task description, a task location, the name of a help message for the task, a task action file, and the files associated with the task. Below are descriptions of the information you must provide in each field.

Task Name	The name of the new task (as it should appear in the lefthand column of the screen). This field has a maximum length of 16 characters and should consist of alphanumeric characters.
Task Description	A description of the new task (as it should appear in the righthand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except an at sign (`@`), circumflex (`^`), tilde (`~`), back quote (`` ` ``), single quote (`'`), and double quotes (`"`).
Task Location	The location of the task in the menu hierarchy, expressed as a pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this task. Each menu name must be separated by colons. For example, the task location for a task entry being added to the `Applications` menu is `main:applications`. Do not include the task name in this location definition. The complete path to this task entry will be the task location plus the task name defined at the first prompt. This field has a maximum length of 50 characters and should consist of alphanumeric characters.
Task Help File Name	Pathname to the item help file for this task entry. If it resides in the directory from which you invoked `edsysadm`, you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by `$EDITOR`) to create one. The new file is created in the current directory and named `Help`.

Task Action	The FACE form name or executable that will be run when this task is selected. This field has a maximum length of 50 characters and should consist of alphanumeric characters. This pathname can be relative to the current directory as well as absolute. (Refer to the *Programmer's Guide: System Services and Application Packaging Tools* for details on the task action file.)
Task Files	Any FACE objects or other executables that support the task action listed above and might be called from within that action. Do not include the item help filename or the task action in this list. Pathnames can be relative to the current directory as well as absolute. A dot (.) implies "all files in the current directory" and includes files in subdirectories. This field is multi-lined and has a maximum length of 50 characters per line. Entries should consist of alphanumeric characters.

The following screen shows a sample task definition form that has been filled in:



```
              Define A Task

Name:  msvrtask
Description:  Task Description
Location:  main:applications:msvr
Help Message:  Help
Action:  form.msvrtask
Task Files:  form.task2, text.task2
```

Deleting a Menu or Task Entry

To delete either a menu or task entry from the interface, execute

```
	delsysadm  name
```

where *name* is the location of the task or menu in the interface, followed by the menu or task name. For example, to delete a task named `mytask` with the location `main:application:mymenu`, execute

```
	delsysadm  main:application:mymenu:mytask
```

Before an entry for a menu can be removed, that menu must be empty (contain no submenus or tasks). If it is not, you must use the `-r` option with `delsysadm`. This option requests that, in addition to the named menu, all submenus and tasks located under that menu be removed. For example, to remove `main:application:mymenu` and all submenus and tasks that reside under it, execute

```
	delsysadm  -r  main:application:mymenu
```

When you use the `-r` option, `delsysadm` checks for dependencies before removing any subentries. (A dependency exists if the menu being removed contains an entry placed there by an application package.) If a dependency is found, you are shown a list of packages that depend on the menu you want to delete and asked whether you you want to continue. If you answer yes, the menu and all of its menus and tasks are removed (even those shown to have dependencies). If you answer no, the menu is not deleted.

CAUTION Use `delsysadm` to remove only those menu or task entries that you have added to the interface with `edsysadm`.

E Error Messages

Introduction

UNIX system error messages are divided into three severity classes: `NOTICE`, `WARNING`, and `PANIC`. When an error message is displayed, its severity class is displayed as the first part of the message. The following UNIX system error message tables are divided into severity classes. A description of each severity class is given with each table.

UNIX System NOTICE Messages

NOTICE error messages provide information on the system status, thus helping you anticipate problems before they occur. These messages are listed alphabetically in the following table.

Error Message (Prefaced by NOTICE)	Description/Action
`anon_resv - Insufficient memory to ...`	A request for memory was denied because memory was overcommitted. If the notice occurs often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`bad block on floppy drive, slice # [or] bad block on integral hard disk diver #, partition #`	An out of range block number was specified for the file system indicated. Check the file system (see Chapter 5, File System Administration).
`bad count on floppy drive, slice # [or] bad count on integral hard disk #, partition #`	The block count in the super block is inconsistent for the file system indicated. Check the file system (see Chapter 5, File System Administration).
`Bad free block count on integral hard disk drive #, partition #.`	The free list block count is inconsistent for the file system indicated by the drive/partition numbers. Check the file system (see Chapter 5, File System Administration).
`Bad free count on floppy drive, slice #.`	The free list block count is inconsistent. Check the file system (see Chapter 5, File System Administration).
`Can't allocate message buffer.`	At initialization time, the IPC facility could not obtain the memory needed for message queues. IPC messages are currently unusable. If repeatable, reconfigure the system to reduce this amount (MSGSEG * MSGSSZ, in the `master.d/msg` file); or, add more physical memory.
`Configured value of NGROUPS_MAX (#) is greater than max (#), NGROUPS_MAX set to #`	The value of NGROUPS_MAX (in the `master.d/kernel` file) exceeds the system-imposed maximum. No immediate action required, the system maximum shall be used. To prevent the notice on future reboots, reconfigure your system with a valid NGROUPS_MAX setting.

Error Message (Prefaced by `NOTICE`)	Description/Action
`Configured value of NGROUPS_MAX (#) is less than min (#), NGROUPS_MAX set to #`	The value of NGROUPS_MAX (in the `master.d/kernel` file) is less then the system-imposed minimum. No immediate action required, the system minimum shall be used. To prevent the notice on future reboots, reconfigure your system with a valid NGROUPS_MAX setting.
`dnlc_purge1: no entries to purge`	All directory name cache entries are in use, thus all names shall not be cached. (This is additional information that may sometimes precede messages of the form ``` ``NOTICE: iget - inode table overflow.'' ```) If the former notice recurs frequently, reconfigure the system with a larger *ncsize* setting (in the `master.d/kernel` file).
`Floppy Access Error: Consult the Error Message Section of the System Administration Utilities Guide`	Verify the diskette is in the drive and the door is closed. Or, the file system on the diskette runs off the end of the disk; reconfigure the file system to be within the boundaries of the diskette. Or, the diskette may be defective; if reformatting fails, replace the diskette.
`hat_ptalloc - Insufficient memory ... [or] hat_sdtalloc - Insufficient memory ...`	A request for memory was denied because memory was overcommitted. If the notice occurs often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`ialloc: inode was already allocated`	A race condition for an inode was avoided (by allocating a different inode). No action required.
`KERNEL: Could not allocate a file table entry`	The system tried to dynamically create a file table entry but failed because memory was low. (At user level the affected process receives *errno* ENFILE, and should retry.) This normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).

Error Message (Prefaced by NOTICE)	Description/Action
`locore access: pid: # pc: 0x# address: 0x# COMMAND::`*command*	The specified command is attempting to dereference a null pointer. This is usually a command bug.
`No memory for name cache`	At initialization time there was insufficient memory for the directory name cache. The system will function properly without the name cache (although if there is so little memory it is likely that something else will malfunction soon after). This normally indicates insufficient memory to support the configured kernel. Consider adding more physical memory.
`no space on floppy drive, slice # [or] no space on integral hard disk drive #, partition #`	The file system indicated has reached its capacity. Reclaim space by deleting obsolete files. Repartition if space continues to be a problem (see Chapter 5, File System Administration).
`Out of inodes on integral hard disk drive #, partition #.`	The indicated file system is out of inodes. Reclaim inodes by deleting obsolete files. If inode outage continues to be a problem, back up the file system, remake the file system with a larger number of inodes, and then restore the file system (see Chapter 5, File System Administration).
`READ CLOCK -- TOO MANY TRIES`	Attempts to read the real time clock failed during system startup. Try resetting the system time using the procedures outlined in Chapter 16, System Setup. If a problem still exists, check the battery. If an error condition still exists the problem may be in the clock hardware; run diagnostics on the time-of-day-clock (see Chapter 4, Diagnostics).
`shmctl - couldn't lock # pages into memory`	An IPC shared memory operation failed because memory was overcommitted. (At user level the affected process receives *errno* ENOMEM, and should retry the system call.) If the notices occur often, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).

Error Message (Prefaced by `NOTICE`)	Description/Action
`spurious iu counter interrupt`	The console or contty driver received an unexpected interrupt. No action needed. If these notices recur frequently suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`stray interrupt at #`	An unexpected interrupt occurred. No action needed. If these notices recur frequently suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`XT: ... no streams ...`	A request for STREAMS by the windowing terminal driver failed because memory was overcommitted. (No immediate action needed, XT sleeps waiting for memory.) If these notices occur often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).

UNIX System WARNING Messages

WARNING error messages indicate that the system may stop functioning if corrective action is not taken. These error messages are listed alphabetically in the following table.

Error Message (Prefaced by WARNING)	Description/Action
`bufcall: could not allocate stream event`	No virtual address space is available from the kernel memory allocator (resulting in a STREAMS operation to fail). If the warning occurs often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`Can not allocate memory for mux_edge`	No virtual address space is available from the kernel memory allocator (resulting in a STREAMS operation to fail). If the warning occurs often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`dirremove: ino #, dev #, nlink #`	The link count for the identified inode was invalid (less than zero). No action required. Link count is set to zero automatically.
`esbbcall: could not allocate stream event`	No virtual address space is available from the kernel memory allocator (resulting in a STREAMS operation to fail). If the warning occurs often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`fifogetid(): could not establish a unique node id`	The kernel was not able to establish a unique node id for a pipe (the system limit is currently in-use). As such, the node id has been set to 0. Operations that rely on the uniqueness of node id's may fail unpredictably. Rebooting is recommended.

Error Message (Prefaced by WARNING)	Description/Action
`floppy disk timeout; request flushed`	Verify the diskette is in the drive and the door is closed. Contact your service representative if the problem recurs.
`floppy disk: Bad address returned from VTOP`	A bad address was passed to the driver or a Virtual TO Physical (VTOP) address conversion failed.
`fs_vcode: vcode overflow`	An RFS implementation limit has been reached. RFS should be stopped and restarted.
`hard disk: Bad sanity word` `hard disk: Cannot read ... on drive #`	The disk VTOC, defect table or sanity track on the indicated drive needs to be rebuilt. The disk is currently unusable. Contact your service representative for assistance or try restoring the disk from backup.
`hard disk: cannot access sector #, head #, cylinder #, on drive #`	A bad disk block was encountered. If the system is in multi-user mode, the disk error daemon should report that the bad block was logged. Otherwise, use **hdefix**(1M) to log the block (see "Bad Block Handling" in Chapter 4, Diagnostics).
`hard disk: cannot recal drive #`	The disk driver could not seek on the indicated drive. Probably a hardware problem; run diagnostics on the drive (see Chapter 4, Diagnostics).
`hard disk: Drive # is in the 1.0 layout.`	The drive is in an old format and cannot be used until converted to the current layout. Contact your service representative for assistance.
`hard disk: drive # out of service`	Probably a hardware problem. Check cabling. Run diagnostics on the drive.
`hard disk: partition # on drive # is marked read only`	If you wish to write to this disk partition, the permissions in the VTOC must be changed. Use **fmthard**(1M).
`hard disk: too little space allocated in driver for defect table on drive #`	Too little space is configured in the disk driver to hold the hard disk defect table. The operating system needs to be rebuilt with a larger `IDDEFSZ` (in `sys/id.h`) and corresponding increase for `iddefect[IDDEFSIZ]` (in the `master.d/idisk` file).

Error Message (Prefaced by `WARNING`)	Description/Action
`HDE queue full, following report not logged ...`	The hard disk error logger is full. No more errors can be logged. Save the unlogged report and add it manually to the disk error log. For more information, see "Bad Block Handling" in Chapter 4, Diagnostics.
`hdeeqd: major(ddev) = # (>=cdevcnt)`	A bad disk block was encountered. If the system is in multi-user mode, the disk error daemon should report that the bad block was logged. Otherwise, use **hdefix**(1M) to log the block. For more information see "Bad Block Handling" in Chapter 4, Diagnostics.
`ID didn't get IDSEEK0, IDSEEK1, or IDXFER`	The integral disk driver received an interrupt that contained unexpected status information.
`IDRECAL called # times for drive #`	The disk driver could not seek on the indicated drive. Possibly a hardware problem; run diagnostics on the drive (see Chapter 4, Diagnostics).
`iget - inode table overflow`	Out of in-core inodes; the number of in-use files exceeds the system configuration. If this problem recurs, reconfigure the system with a larger in-core inode table (e.g., for the S5 file system increase **NINODE** in the **master.d/s5** file).
`In-Core Disk not existing`	There is no valid VTOC on the In-Core Disk. Boot the Essential Boot Utilities floppies again. Contact your service representative if the problem recurs. This problem can only occur when initially installing the system.
`insq: attempt to insert message out of order on q #`	An attempt has been made to insert a STREAMS message onto a queue and the priority band associated with the message does not belong where it is attempting to be put. Probable bug in the system code attempting to do the insert.
`iu_get_buffer: out of blocks`	A request for STREAMS by the console driver failed because memory was overcommitted. If the warnings occur often, try rebooting. If the problem recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).

Error Message (Prefaced by `WARNING`)	Description/Action
`KERNEL: munlink: could not perform unlink ioctl, closing anyway`	The ioctl to unlink a STREAMS multiplexor failed. The close of the device will take place anyhow. No action required.
`ldterm: ... out of blocks` [or] `ldtermrsrv: out of blocks`	A request for STREAMS by the terminal line discipline failed because memory was overcommitted. If the warnings occur often, try rebooting. If the problem recurs this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`ldtermopen: open fails, can't allocate state structure`	The terminal line discipline *ldterm* could not be started because a request for memory (STREAMS) failed. Affected lines shall have only "raw" processing performed (e.g., typed characters won't appear on the terminal until control-j is typed). Try restarting the handler on the affected lines using `strchg`(1M). If the warning recurs, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`Max system class priority must be >= 39, configured value is # - resetting v.v_maxsyspri to 39`	The `MAXCLSYSPRI` tunable (in the `master.d/kernel` file) is less than the system-imposed minimum of 39. The system minimum setting shall be used. Review your scheduler configuration and reconfigure as needed.
`missing exec capability for` *string*	The boot module named by *string* is not in a recognizable object file format. Either the file is corrupted: restore the file from backup and rebuild the system, or reinstall the add-on package that includes *string*; or, the file erroneously resides in the boot directory: exclude or move the file and rebuild the system.
`nmgetid(): could not establish a unique node id`	The kernel was not able to establish a unique node id for a named stream (the system limit is currently in-use). As such, the node id has been set to 0. Operations that rely on the uniqueness of node id's may fail unpredictably. Rebooting is recommended.

Error Message (Prefaced by WARNING)	Description/Action
`No ptem structures`	There is an insufficient number of pseudo-terminals configured on your system. Increase the number of pseudo-terminals configured.
`Page frame 0x# locked permanently`	Self-explanatory. A page is locked a number of times which is greater than the variable can hold.
`PORTS: EXPRESS QUEUE OVERLOAD: One entry lost`	The number of save queues in the PORTS tty subsystem may be insufficient. If warning occur often, reconfigure the system with a larger **SAVEXP** value (in the **master.d/ports** file). If the warning recurs, suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`PORTS: FAULT - opcode=#, board=#`	An invalid PORTS opcode was encountered. A firmware problem is indicated. Try resetting the the board using the **pump**(1M) command. If a problem still exists reboot the system. If the warning recurs, suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`PORTS: Hardware error board #, port #. Can't get buffer`	Log the error message. Reboot the system. Contact your service representative if the problem recurs.
`PORTS: QFAULT - opcode=#, board=#`	The job queue for the specified PORTS board is invalid. A firmware problem is indicated. Try resetting the the board using the **pump**(1M) command. If a problem still exists reboot the system. If the warning recurs, suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`PORTS: unknown completion code: #`	Try resetting the the board using the **pump**(1M) command. If a problem still exists reboot the system. If the problem recurs suspect a hardware problem; perform diagnostics (see Chapter 4, Diagnostics).
`prinit: can't get unique device number`	At system startup the system was unable to find an unused device number for internal use. The system should continue to function properly.

Error Message (Prefaced by WARNING)	Description/Action
`rmfree map overflow # Lost # items`	Ran out of entries used to manage the available memory for the indicated map. The map overflow # is the address of the table that is too small; the commands "`nm -x /stand/unix | grep #`" can be used to get the name of the map table that overflowed. If the warnings persist, reconfigure the system with a larger table for the indicated map. (The system currently has three map tables, whose sizes are determined by the **SPTMAP**, **MSGMAP** and **SEMMAP** tunables; see Chapter 8, Performance Management.)
`setqback: can't allocate qband` [or] `stropen: out of queues` [or] `strrput: could not allocate qband`	No virtual address space is available from the kernel memory allocator (causing the named STREAMS operation to fail). If the warning occurs often, try rebooting. If the problem repeats, this normally indicates insufficient memory to support the configured kernel and/or system load. Options: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`Sorry, pid #` *(process)* `was killed due to lack of swap space`	The system could not start the named process because swap space would then be overcommitted. Try again. If the situation occurs often, increase swap space using the **swap**(1M) command; or, run fewer simultaneous processes.
`swapadd: configured values for swplo (#) and nswap (#) are too large for partition size (#). nswap reduced to # blocks`	The swap area specified by **SWAPDEV** (in the **master.d/kernel** file) is larger than the swap partition. No immediate action required; a valid size has been calculated and used. To avoid this notice on future reboots, correct **SWAPDEV**, rebuild and reboot.
`swapdel - too few free pages`	A request to delete a swap file was denied; removing this swap area would have reduced the available swap below the minimum required in the system.
`swith(): default #`	An internal consistency check failed in the IPC shared memory subsystem.
`unreadable CRC hard disk error: maj/min = #/# block = #`	A bad disk block was encountered. If the system is in multi-user mode, the disk error daemon should report that the bad block was logged. Otherwise, use **hdefix**(1M) to log the block. For more information, see "Bad Block Handling" in Chapter 4, Diagnostics.

UNIX System PANIC Messages

PANIC error messages indicate a problem severe enough that the UNIX operating system must stop. The cause is usually a hardware problem or a problem in the kernel software. Some file systems may be corrupted, but the UNIX system checks for this when it is restarted. As with most sophisticated computer systems, panics will occasionally occur; they should not cause much concern. If a particular panic occurs repeatedly (or predictably), you should contact your service representative. These error messages are listed alphabetically in the following table.

NOTE After the panic completes, take a system dump if desired (see "Recovering from System Trouble" in the "Diagnostics" chapter). If panics occur frequently, contact your service representative.

Error Message (Prefaced by PANIC)	Description/Action
`as_ctl: bad operation #`	An internal memory management routine (**as_ctl**) was called with an invalid argument. Probable bug in the system code making the call.
`blkdev`	The major device number of a block device exceeds the number of such drivers generated into the system. Verify the configuration information for any new or recently changed device drivers.
`bp_map - non B_PAGEIO` `bp_mapin`	The named routine was called to operate on a page but the page had inappropriate flags set.
`Call to internal routine of uninstalled package`	A routine was called that is part of an optional and currently noninstalled package.
`Can't allocate memory for dispatcher queues` `Can't allocate memory for dispq active map`	During system startup, couldn't allocate the memory required for the process scheduler data structures. Normally this indicates insufficient memory to support the configured kernel. Normally occurs only after reducing equipped memory or after reconfiguring for add-on software or tunable settings. Try adding physical memory; or, reboot a working unix, review any recent changes and reconfigure as needed (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management).
`cannot allocate segkmap`	An unrecoverable fault occurred during system startup.
`cannot allocate segu`	An unrecoverable fault occurred during system startup.
`chgstropts: can't allocate stroptions message`	A request for memory by the terminal line discipline failed at a critical point. Normally this indicates insufficient memory to support the configured kernel and/or system load. If the problem recurs: add more physical memory; reduce system load; or, configure a smaller OS (exclude unused modules, adjust tunable settings).
`Could not allocate space for mux_nodes`	During system startup, no virtual address space is available from the kernel memory allocator (resulting in the failure of STREAMS to allocate critical data structures). Normally this indicates insufficient

Error Message (Prefaced by `PANIC`)	Description/Action
	memory to support the configured kernel. Normally occurs only after reducing equipped memory or after reconfiguring for add-on software or tunable settings. Try adding physical memory; or, reboot a working unix, review any recent changes and reconfigure as needed (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management).
`devtab`	The list header for the chain of buffers attached to the block-type device cannot be found. Verify the configuration information for any new or recently changed device drivers.
`fbread`	An internal consistency check failed; the number of bytes requested for a block read exceeded the maximum block size.
`floppy disk: bad address returned by/from VTOP`	A bad address was passed to the driver or a Virtual TO Physical (VTOP) address conversion failed.
`hard disk: bad address returned by VTOP`	same as for floppy disk.
`hat_`*string* `...`	An unrecoverable error was detected by the named Hardware Address Translation (HAT) routine, `hat_`*string*().
`ifint`	The floppy driver received an interrupt that contained unexpected status information.
`Illegal or unconfigured class (`*class*`) specified for init process`	The `INITCLASS` tunable (in the `master.d/kernel` file) specifies an invalid scheduler class. (On startup only.) Reboot a working `unix` (see "Recovering from an Unbootable Operating System" in Chapter 8) and reconfigure the system with a valid `INITCLASS` setting (see Chapter 10, Process Scheduling).

Error Message (Prefaced by PANIC)	Description/Action
`Illegal SIT counter selected`	An illegal System Interval Timer (SIT) counter was selected; (the kernel function that handles SIT requests was called internally with a bad argument). A fault in (add-on) system software is implicated.
`Init process cannot enter ... scheduling class.`	A problem was encountered placing the init process in the scheduling class specified by the **INITCLASS** tunable (in the **master.d/kernel** file). (On startup only.) Reboot a working unix (see "Recovering from an Unbootable Operating System" in Chapter 8) and review the scheduler configuration to verify that **INITCLASS** specifies a properly configured class (see Chapter 10, Process Scheduling); reconfigure the system as needed.
`inoinit: no memory for inodes`	During system startup, couldn't allocate the memory needed for the inode table. Normally this indicates insufficient memory to support the configured kernel. Normally occurs only after reducing equipped memory or after reconfiguring for add-on software or tunable settings. Try adding physical memory; or, reboot a working unix, review any recent changes and reconfigure as needed (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management).
`iutimefn`	An internal consistency check failed in the console driver.
`KERNEL BUS TIMEOUT`	A bus request was not fulfilled within the allotted time. Normally a hardware fault.
`KERNEL DATA ALIGNMENT ERROR`	The system software attempted to execute a malaligned instruction or access malaligned data (e.g., execution was attempted using an odd number as a pointer to a 16-bit quantity or a number that is not a multiple of 4 as a pointer to a 32-bit quantity). Normally a software fault.
`KERNEL MMU FAULT #`	A bus request was not fulfilled within the allotted time.

Error Message (Prefaced by PANIC)	Description/Action
`KERNEL MMU FAULT ...`	The Memory Management Unit (MMU) acknowledged a fault during the execution of an instruction while in kernel mode.
`KERNEL MODE ... FAULT`	The processor unexpectedly registered the error given by ...
`KERNEL MODE FAULT, FT=#, ISC=#`	The processor unexpectedly registered an error, identified by Fault Type (FT) and Internal State Code (ISC).
`kernel process stack exception, ISC=#.`	A stack reference caused a memory fault.
`kmem_coalesce ... [or] kmem_free ...`	An internal consistency check failed in the Kernel Memory Allocator (KMA). Probable bug in system software misusing the KMA (e.g., writing memory after having returning same to the KMA).
`kmem_init: failed to allocate big [small] pool`	At initialization time, the Kernel Memory Allocator (KMA) could not obtain the initial memory to be managed by KMA because memory was already over-committed. Normally this indicates insufficient memory to support the configured kernel. Normally occurs only after reducing equipped memory or after reconfiguring for add-on software or tunable settings. Try adding physical memory; or, reboot a working unix, review any recent changes and reconfigure as needed (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management).
`lock_again: ...`	An internal consistency check failed in the IPC shared memory subsystem.
`main: copyout of icode failed`	The kernel was not able to copy the assembly code that is used to start up the system.
`main: return from` *string*	The kernel daemon identified by *string* unexpectedly died.

Error Message (Prefaced by PANIC)	Description/Action
`newproc - fork failed [or]` `newproc - no procs`	The kernel ran out of process table slots while creating kernel processes at boot time. Reboot a working unix (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management), and check the value of **NPROC** (in the **master.d/kernel** file).
`no page`	A page that was marked locked in memory could not be found.
`No space for mapping files`	An unrecoverable fault occurred during system startup.
`No space for mapping u areas`	An unrecoverable fault occurred during system startup.
page_*string* ...	An unrecoverable error was detected by the named page management routine, page_*string*().
`physio unlock`	Unable to unlock a part of physical memory.
`process 0 - creation failed`	An unrecoverable fault occurred during system startup.
`process exception, proc = 0x#, psw= 0x#, pcbp = 0x#.`	The system took a process exception while in the kernel or an interrupt handler. Suspect a hardware problem if the error recurs.
`pvn_vptrunc zbytes`	File truncate operation called with an illegal size (exceeded maximum system block size).
`s5mountroot: no memory for VFS private data`	Couldn't allocate the memory needed for the Virtual File System (VFS) private data allocation at system startup. Normally this indicates insufficient memory to support the configured kernel. Normally occurs only after reducing equipped memory or after reconfiguring for add-on software or tunable settings. Try adding physical memory; or, reboot a working unix, review any recent changes and reconfigure as needed (see "Recovering from an Unbootable Operating System" in Chapter 8, Performance Management).
`s5unmap: fewer than 0 mappings`	The number of VM mappings of a file went below zero.

Error Message (Prefaced by PANIC)	Description/Action
s5_getapage page_enter	The kernel was attempting to add a new page to the page cache but the page was found to be in the cache already.
segmap_create segkmap	An unrecoverable fault occurred during system startup.
segu_create segu	An unrecoverable fault occurred during system startup.
seg_*string* [or] segx_*string* ...	An unrecoverable error was detected by the named segment management routine.
shm_unlock: ...	An internal consistency check failed in the IPC shared memory subsystem.
strsendsig: unknown event #	An internal consistency check failed in the STREAMS subsystem. A STREAMS event contains an undefined event type.
swapconf - swapadd failed	Unable to add the initial swap file. Check the **SWAPDEV** parameter (in the **master.d/kernel** file) and reconfigure as needed.
swapconf lookupname *pathname* failed - error #	Could not translate the swap pathname to a vnode for the initial swap file. Verify that the swap file exists; check **SWAPDEV** parameter (in the **master.d/kernel** file) and reconfigure as needed.
swap_free	The swap area data structure is corrupted or a bad pointer is passed.
SYSTEM PARITY ERROR INTERRUPT	A memory parity error occurred. If this occurs repeatedly, a hardware problem exists.
Timeout table overflow	The queue for timeout requests overflowed while attempting to add another entry. If repeatable or to help prevent re-occurrence, reconfigure the system with a larger **NCALL** setting (in the **master.d/kernel** file).

Error Message (Prefaced by `PANIC`)	Description/Action
`Unexpected user stack fault, ISC = #`	User stack fault occurred which was neither a stack bound or page fault. Probably due to an interrupt vector ID fetch fault. Check interrupt vector table (beginning at virtual location 140) and hardware configuration.
`usrxmemflt: no as allocated.`	The process address space had not been allocated when a user took fault.
`vfs_mountroot: cannot mount root`	At system startup no appropriate root file system could be mounted. The root file system may be corrupted, or the system may have been configured without the necessary file system code.

UNIX System Call Error Messages

A system call that is unsuccessful is indicated by an otherwise impossible value. This impossible value is almost always a −1. (When a system call is successful, a value of 0 is returned to the calling process.) When a system call is unsuccessful, an error number is also made available in the external variable `errno`. `errno` is not cleared on successful calls, so it should be tested only after an error has been indicated.

The following is a list of the error numbers and messages as defined in `<errno.h>`.

Error Number	Error Message	Description
1	EPERM	**Not super-user** Typically this error indicates an attempt to modify a file in some way forbidden except to its owner or the super-user. It is also returned for attempts by ordinary users to do things allowed only to the super-user.
2	ENOENT	**No such file or directory** A file name is specified and the file should exist but doesn't, or one of the directories in a path name does not exist.
3	ESRCH	**No such process** No process can be found corresponding to the that specified by PID in the `kill` or `ptrace` routine.
4	EINTR	**Interrupted system call** An asynchronous signal (such as interrupt or quit), which the user has elected to catch, occurred during a system service routine. If execution is resumed after processing the signal, it will appear as if the interrupted routine call returned this error condition.
5	EIO	**I/O error** Some physical I/O error has occurred. This error may in some cases occur on a call following the one to which it actually applies.
6	ENXIO	**No such device or address** I/O on a special file refers to a subdevice which does not exist, or exists beyond the limit of the device. It may also occur when, for example, a tape drive is not on-line or no disk pack is loaded on a drive.
7	E2BIG	**Arg list too long** An argument list longer than ARG_MAX bytes is presented to a member of the **exec** family of routines. The argument list limit is sum of the size of the argument list plus the size of the environment's exported shell variables.

Error Number	Error Message	Description
8	ENOEXEC	**Exec format error** A request is made to execute a file which, although it has the appropriate permissions, does not start with a valid format (see `a.out`(4)).
9	EBADF	**Bad file number** Either a file descriptor refers to no open file, or a **read** [respectively, **write**] request is made to a file that is open only for writing (respectively, reading).
10	ECHILD	**No child processes** A **wait** routine was executed by a process that had no existing or unwaited-for child processes.
11	EAGAIN	**No more processes** For example, the **fork** routine failed because the system's process table is full or the user is not allowed to create any more processes. Or a system call failed because of insufficient memory or swap space.
12	ENOMEM	**Not enough space** During execution of an **exec**, **brk**, or **sbrk** routine, a program asks for more space than the system is able to supply. This is not a temporary condition; the maximum size is a system parameter. The error may also occur if the arrangement of text, data, and stack segments requires too many segmentation registers, or if there is not enough swap space during the **fork** routine. If this error occurs on a resource associated with Remote File Sharing (RFS), it indicates a memory depletion which may be temporary, dependent on system activity at the time the call was invoked.
13	EACCES	**Permission denied** An attempt was made to access a file in a way forbidden by the protection system.
14	EFAULT	**Bad address** The system encountered a hardware fault in attempting to use an argument of a routine. For example, **errno** potentially may be set to **EFAULT** any time a routine that takes a pointer argument is passed an invalid address, if the system can detect the condition. Because systems will differ in their ability to reliably detect a bad address, on some implementations passing a bad address to a routine will result in undefined behavior.

Error Number	Error Message	Description
15	ENOTBLK	**Block device required** A non-block file was mentioned where a block device was required (e.g., in a call to the mount routine).
16	EBUSY	**Device busy** An attempt was made to mount a device that was already mounted or an attempt was made to dismount a device on which there is an active file (open file, current directory, mounted-on file, active text segment). It will also occur if an attempt is made to enable accounting when it is already enabled. The device or resource is currently unavailable.
17	EEXIST	**File exists** An existing file was mentioned in an inappropriate context (e.g., call to the `link` routine).
18	EXDEV	**Cross-device link** A link to a file on another device was attempted.
19	ENODEV	**No such device** An attempt was made to apply an inappropriate operation to a device (e.g., read a write-only device).
20	ENOTDIR	**Not a directory** A non-directory was specified where a directory is required (e.g., in a path prefix or as an argument to the `chdir` routine).
21	EISDIR	**Is a directory** An attempt was made to write on a directory.
22	EINVAL	**Invalid argument** An invalid argument was specified (e.g., unmounting a non-mounted device, mentioning an undefined signal in a call to the `signal` or `kill` routine. Also set by the functions described in the math package (3M).
23	ENFILE	**File table overflow** The system file table is full (i.e., SYS_OPEN files are open, and temporarily no more files can be opened).
24	EMFILE	**Too many open files** No process may have more than OPEN_MAX file descriptors open at a time.

Error Number	Error Message	Description
25	ENOTTY	**Not a typewriter** A call was made to the ioctl routine specifying a file that is not a special character device.
26	ETXTBSY	**Text file busy** An attempt was made to execute a pure-procedure program that is currently open for writing. Also an attempt to open for writing or to remove a pure-procedure program that is being executed.
27	EFBIG	**File too large** The size of a file exceeded the maximum file size, FCHR_MAX (see getrlimit).
28	ENOSPC	**No space left on device** While writing an ordinary file or creating a directory entry, there is no free space left on the device. In the fcntl routine, the setting or removing of record locks on a file cannot be accomplished because there are no more record entries left on the system.
29	ESPIPE	**Illegal seek** A call to the lseek routine was issued to a pipe.
30	EROFS	**Read-only file system** An attempt to modify a file or directory was made on a device mounted read-only.
31	EMLINK	**Too many links** An attempt to make more than the maximum number of links, LINK_MAX, to a file.
32	EPIPE	**Broken pipe** A write on a pipe for which there is no process to read the data. This condition normally generates a signal; the error is returned if the signal is ignored.
33	EDOM	**Math argument out of domain of func** The argument of a function in the math package (3M) is out of the domain of the function.
34	ERANGE	**Math result not representable** The value of a function in the math package (3M) is not representable within machine precision.
35	ENOMSG	**No message of desired type** An attempt was made to receive a message of a type that does not exist on the specified message queue (see msgop(2)).

Error Number	Error Message	Description
36	EIDRM	**Identifier removed** This error is returned to processes that resume execution due to the removal of an identifier from the file system's name space (see **msgctl**(2), **semctl**(2), and **shmctl**(2)).
37	ECHRNG	**Channel number out of range**
38	EL2NSYNC	**Level 2 not synchronized**
39	EL3HLT	**Level 3 halted**
40	EL3RST	**Level 3 reset**
41	ELNRNG	**Link number out of range**
42	EUNATCH	**Protocol driver not attached**
43	ENOCSI	**No CSI structure available**
44	EL2HLT	**Level 2 halted**
45	EDEADLK	**Deadlock condition** A deadlock situation was detected and avoided. This error pertains to file and record locking.
46	ENOLCK	**No record locks available** There are no more locks available. The system lock table is full (see **fcntl**(2)).
60	ENOSTR	**Device not a stream** A **putmsg** or **getmsg** system call was attempted on a file descriptor that is not a STREAMS device.
61	ENODATA	**No data available**
62	ETIME	**Timer expired** The timer set for a STREAMS `ioctl` call has expired. The cause of this error is device specific and could indicate either a hardware or software failure, or perhaps a timeout value that is too short for the specific operation. The status of the `ioctl` operation is indeterminate.

Error Number	Error Message	Description
63	ENOSR	**Out of stream resources** During a STREAMS open, either no STREAMS queues or no STREAMS head data structures were available. This is a temporary condition; one may recover from it if other processes release resources.
64	ENONET	**Machine is not on the network** This error is Remote File Sharing (RFS) specific. It occurs when users try to advertise, unadvertise, mount, or unmount remote resources while the machine has not done the proper startup to connect to the network.
65	ENOPKG	**Package not installed** This error occurs when users attempt to use a system call from a package which has not been installed.
66	EREMOTE	**Object is remote** This error is RFS specific. It occurs when users try to advertise a resource which is not on the local machine, or try to mount/unmount a device (or pathname) that is on a remote machine.
67	ENOLINK	**Link has been severed** This error is RFS specific. It occurs when the link (virtual circuit) connecting to a remote machine is gone.
68	EADV	**Advertise error** This error is RFS specific. It occurs when users try to advertise a resource which has been advertised already, or try to stop the RFS while there are resources still advertised, or try to force unmount a resource when it is still advertised.
69	ESRMNT	**Srmount error** This error is RFS specific. It occurs when an attempt is made to stop RFS while resources are still mounted by remote machines, or when a resource is readvertised with a client list that does not include a remote machine that currently has the resource mounted.
70	ECOMM	**Communication error on send** This error is RFS specific. It occurs when the current process is waiting for a message from a remote machine, and the virtual circuit fails.
71	EPROTO	**Protocol error** Some protocol error occurred. This error is device specific, but is generally not related to a hardware failure.

Error Number	Error Message	Description
74	EMULTIHOP	**Multihop attempted** This error is RFS specific. It occurs when users try to access remote resources which are not directly accessible.
76	EDOTDOT	**Error 76** This error is RFS specific. A way for the server to tell the client that a process has transferred back from mount point.
77	EBADMSG	**Not a data message** During a **read**, **getmsg**, or ioctl I_RECVFD system call to a STREAMS device, something has come to the head of the queue that can't be processed. That something depends on the system call: **read**: control information or a passed file descriptor. **getmsg**: passed file descriptor. ioctl: control or data information.
78	ENAMETOOLONG	**File name too long** The length of the path argument exceeds PATH_MAX, or the length of a path component exceeds NAME_MAX while _POSIX_NO_TRUNC is in effect; see limits(4).
79	EOVERFLOW	**Error 79** Value too large to be stored in data type.
80	ENOTUNIQ	**Name not unique on network** Given log name not unique.
81	EBADFD	**File descriptor in bad state** Either a file descriptor refers to no open file or a read request was made to a file that is open only for writing.
82	EREMCHG	**Remote address changed**
83	ELIBACC	**Cannot access a needed shared library** Trying to **exec** an a.out that requires a shared library and the shared library doesn't exist or the user doesn't have permission to use it.
84	ELIBBAD	**Accessing a corrupted shared library** Trying to **exec** an a.out that requires a shared library (to be linked in) and **exec** could not load the shared library. The shared library is probably corrupted.
85	ELIBSCN	**.lib section in a.out corrupted** Trying to **exec** an a.out that requires a shared library (to be linked in) and there

Error Number	Error Message	Description
		was erroneous data in the `.lib` section of the `a.out`. The `.lib` section tells **exec** what shared libraries are needed. The `a.out` is probably corrupted.
86	ELIBMAX	**Attempting to link in more shared libraries than system limit** Trying to **exec** an `a.out` that requires more static shared libraries than is allowed on the current configuration of the system. See the *System Administrator's Guide*.
87	ELIBEXEC	**Cannot exec a shared library directly** Attempting to **exec** a shared library directly.
88	EILSEQ	**Error 88** Illegal byte sequence. Handle multiple characters as a single character.
89	ENOSYS	**Operation not applicable**
90	ELOOP	**Number of symbolic links encountered during path name traversal exceeds MAXSYMLINKS**
91	ERESTART	**Error 91** Interrupted system call should be restarted.
92	ESTRPIPE	**Error 92** Streams pipe error (not externally visible).
93	ENOTEMPTY	**Directory not empty**
94	EUSERS	**Too many users** Too many users.
95	ENOTSOCK	**Socket operation on non-socket** Self-explanatory.
96	EDESTADDRREQ	**Destination address required** A required address was omitted from an operation on a transport endpoint. Destination address required.
97	EMSGSIZE	**Message too long** A message sent on a transport provider was larger than the internal message buffer or some other network limit.
98	EPROTOTYPE	**Protocol wrong type for socket** A protocol was specified that does not support the semantics of the socket type requested.

Error Number	Error Message	Description
99	ENOPROTOOPT	**Protocol not available** A bad option or level was specified when getting or setting options for a protocol.
120	EPROTONOSUPPORT	**Protocol not supported** The protocol has not been configured into the system or no implementation for it exists.
121	ESOCKTNOSUPPORT	**Socket type not supported** The support for the socket type has not been configured into the system or no implementation for it exists.
122	EOPNOTSUPP	**Operation not supported on transport endpoint** For example, trying to accept a connection on a datagram transport endpoint.
123	EPFNOSUPPORT	**Protocol family not supported** The protocol family has not been configured into the system or no implementation for it exists. Used for the Internet protocols.
124	EAFNOSUPPORT	**Address family not supported by protocol family** An address incompatible with the requested protocol was used.
125	EADDRINUSE	**Address already in use** User attempted to use an address already in use, and the protocol does not allow this.
126	EADDRNOTAVAIL	**Cannot assign requested address** Results from an attempt to create a transport endpoint with an address not on the current machine.
127	ENETDOWN	**Network is down** Operation encountered a dead network.
128	ENETUNREACH	**Network is unreachable** Operation was attempted to an unreachable network.
129	ENETRESET	**Network dropped connection because of reset** The host you were connected to crashed and rebooted.
130	ECONNABORTED	**Software caused connection abort** A connection abort was caused internal to your host machine.
131	ECONNRESET	**Connection reset by peer** A connection was forcibly closed by a peer. This normally results from a loss of the connection on the remote host due to a timeout or a reboot.

Error Number	Error Message	Description
132	ENOBUFS	**No buffer space available** An operation on a transport endpoint or pipe was not performed because the system lacked sufficient buffer space or because a queue was full.
133	EISCONN	**Transport endpoint is already connected** A connect request was made on an already connected transport endpoint; or, a *sendto* or *sendmsg* request on a connected transport endpoint specified a destination when already connected.
134	ENOTCONN	**Transport endpoint is not connected** A request to send or receive data was disallowed because the transport endpoint is not connected and (when sending a datagram) no address was supplied.
143	ESHUTDOWN	**Cannot send after transport endpoint shutdown** A request to send data was disallowed because the transport endpoint had already been shut down.
144	ETOOMANYREFS	**Too many references: cannot splice**
145	ETIMEDOUT	**Connection timed out** A connect or send request failed because the connected party did not properly respond after a period of time. (The timeout period is dependent on the communication protocol.)
146	ECONNREFUSED	**Connection refused** No connection could be made because the target machine actively refused it. This usually results from trying to connect to a service that is inactive on the remote host.
147	EHOSTDOWN	**Host is down** A transport provider operation failed because the destination host was down.
148	EHOSTUNREACH	**No route to host** A transport provider operation was attempted to an unreachable host.
19	EALREADY	**Operation already in progress** An operation was attempted on a non-blocking object that already had an operation in progress.
150	EINPROGRESS	**Operation now in progress** An operation that takes a long time to complete (such as a **connect**) was attempted on a non-blocking object.
151	ESTALE	**Stale NFS file handle**

Diagnostic Monitor Program Error Messages

The Diagnostic Monitor (DGMON) program provides the user the ability to execute test phases on the 3B2 Computer. If a problem occurs while using the diagnostic monitor program, an error message is displayed on the console terminal. These error messages are prefaced by `DIAGNOSTIC MONITOR ERROR #`. These error messages are defined numerically in the following table.

Error Number	Error Message	Description/Action
1-00	`FILE SYSTEM IS INACCESSIBLE CONTROL WILL RETURN TO MAINTENANCE CONTROL PROGRAM.`	Retry request. If it fails again, there is a possible problem with the protected disk cylinder where diagnostics reside.
1-01	`UNKNOWN ID CODE (dev code) FOR DEVICE IN SLOT (slot #) NO DIAGNOSTIC TESTS RUN FOR THIS SLOT. CHECK EDT.`	Retry request. If it fails again, the device is not recognized because installation is incomplete or the device reports a bad ID code.
1-02	`CANNOT FIND FILE: (file name) DIAGNOSTIC REQUEST ABORTED.`	Retry request.
1-03	`CANNOT LOAD FILE: (file name) DIAGNOSTIC REQUEST ABORTED.`	Retry request.
1-04	`UNEXPECTED DIAGNOSTIC EXCEPTION. DIAGNOSTIC REQUEST ABORTED.`	Retry request.
1-05	`UNEXPECTED DIAGNOSTIC INTERRUPT. DIAGNOSTIC REQUEST ABORTED.`	Retry request.
1-06	`NON-EXISTENT UNIT: (device name) THE EQUIPPED UNIT TYPES ARE: (list of device names)`	Retry request.
1-07	`INVALID UNIT NUMBER FOR (device name). THE EQUIPPED UNITS ARE: (list of device names) RETRY REQUEST.`	Retry request.
1-08	`(echo of input string) UNRECOGNIZABLE DIAGNOSTIC REQUEST. CHECK REQUEST SYNTAX AND RE-ENTER.`	Retry request.

Error Number	Error Message	Description/Action
1-09	INVALID REPEAT VALUE RE-ENTER REQUEST USING VALUE BETWEEN 1 AND 65536.	Retry request.
1-10	INVALID PHASE(S) REQUESTED. CHECK REQUESTED PHASE TABLE AND RETRY.	Retry request.
1-11	REDUNDANT DIAGNOSTIC REQUEST OPTION. RE-ENTER REQUEST.	Retry request.
1-12	SOAK AND UCL ARE INCOMPATIBLE DIAGNOSTIC OPTIONS. RE-ENTER REQUEST, OMITTING ONE.	Retry request.
1-13	UNIT OR UNIT TYPE NEEDED FOR PHASE OPTION REQUEST. RE-ENTER REQUEST.	Retry request.
1-14	USE UNIT TYPE ONLY FOR PHASE DISPLAY REQUEST. RE-ENTER REQUEST.	Retry request.

Equipped Device Table Completion Error Messages

The filledt program is a disk-based routine that runs at the firmware level. The filledt program completes the system equipped device table. These error messages are output if trouble occurs while filledt is executing. All errors except 1-08 and 1-09 are suppressed during autoboot. When manually booting the system, no errors are suppressed. Each error message is prefaced by EDT COMPLETION ERROR #. The EDT completion error messages are defined numerically in the following table.

Error Number	Error Message	Description/Action
1-00	FILE SYSTEM IS INACCESSIBLE. CONTROL WILL RETURN TO MAINTE-NANCE CONTROL PROGRAM.	Retry request. If it fails again, there is a possible problem with the protected disk cylinder where diagnostics reside.
1-01	ERROR OCCURRED DURING SYSTEM CONFIGURATION. CONSOLE LOCATION PROCEEDING. CHECK EDT.	Check equipped device table, device entry garbled.
1-02	CANNOT FIND FILE: (file name) REQUEST ABORTED.	Retry request. Contact your service representative.
1-03	CANNOT LOAD FILE: (file name) REQUEST ABORTED.	Retry request. Contact your service representative.
1-04	UNEXPECTED EXCEPTION REQUEST ABORTED.	Retry request. Contact your service representative.
1-05	UNEXPECTED INTERRUPT REQUEST ABORTED.	Retry request. Contact your service representative.
1-06	SYSGEN FAILED FOR (device name) IN SLOT (slot #) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.	The peripheral device is not responding to configuration requests. Retry request. Contact your service representative.
1-07	DSD FAILED FOR (device name) IN SLOT (slot #) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.	The peripheral device is not responding to configuration requests. Retry request. Contact your service representative.

Error Number	Error Message	Description/Action
1-08	`UNKNOWN ID CODE (id code) IN SLOT (slot #) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.`	Device installation is incomplete or the device has failed. Retry request. Contact your service representative if installation is not in process.
1-09	`UNKNOWN SUBDEVICE ID CODE FOR DEVICE (device name) IN SLOT (slot #) EQUIPPED DEVICE TABLE COMPLETION WILL CONTINUE. CHECK EDT.`	Device installation is incomplete or the device has failed. Retry request. Contact your service representative if installation is not in process.
1-10	`EDT EXCEEDS ALLOCATED SPACE AND CANNOT BE COMPLETED. REDUCE SYSTEM CONFIGURATION.`	Remove some devices. Retry request.

Firmware Error Messages

The firmware mode is the state the 3B2 computer must be in for you to interface with several software programs. If a problem occurs while in this state, a firmware error message is displayed on the console terminal. These error messages are prefaced by `FW ERROR #`. These error messages are defined numerically in the following table.

Error Number	Error Message	Description/Action
1-01	`NVRAM SANITY FAILURE`	Try system power-up again. If same message appears, check the battery.
1-02	`DISK SANITY FAILURE`	Contact your service representative.
1-03	`UNEXPECTED FAULT`	Contact your service representative.
1-04	`UNEXPECTED INTERRUPT`	Contact your service representative.
1-05	`SELF-CONFIGURATION FAILURE UNEXPECTED INTERRUPT`	Contact your service representative.
1-06	`BOOT FAILURE`	If diskette boot, ensure that correct diskette is in the drive.
1-07	`FLOPPY KEY CREATE FAILURE`	Ensure formatted diskette in the drive when go is entered.
1-08	`MEMORY TEST FAILURE`	A failure occurred during pattern tests of the first 256K bytes of the system board memory. Power-cycle the machine to repeat the tests. Contact your service representative if the failure occurs.
1-09	`DISK FORMAT NOT COMPATIBLE WITH SYSTEM`	Message will appear during upgrade to Release 2.0. A 1.0 disk format was detected by firmware. Proceed with the Release 2.0 upgrade procedure. If an upgrade is not in progress, contact your service representative.

Error Number	Error Message	Description/Action

—	`id # CRC error at disk address X`
	
	`if - CRC error at disk address X`
		The # is a decimal number (0 or 1). The hard disk drive is `id`: the integral diskette drive is `if`. The `X` is an 8-character hexadecimal word specifying the physical cylinder number high (pcnh), the physical cylinder number low (pcnl), the physical head number (phn), and the physical sector number (psn). Refer to Chapter 15, "Storage Device Management," for complete information. If the system will boot (run UNIX operating system), add the identified defect to the defect map using the **hdeadd** and **hdefix** command. If you cannot resolve the problem, call your service representative.

Boot and Configuration Error Messages

The error messages in this section are returned when an error occurs during the boot or configuration process. These errors are not fatal, and are recoverable given the behavior of `cunix` when it encounters an error; see `cunix`(1M), the "Machine Management" chapter, and the section "Configuring the UNIX Operating System" in the "Performance Management" chapter, for more details.

Error Message	Description/Action
`flexname too long`	One of the object files being loaded contains a symbol which is more than 256 characters in length. This is a flexname size limit imposed by self-configuration. Either shorten symbol name or change flexname size allowed by self-configuration.
`MAXCNTL exceeded`	Maximum number of controllers allowed per device (16) exceeded. Indicates an illegal hardware configuration. Correct hardware configuration then reboot.
`No memory for EXCLUDE list`	Unable to allocate memory for EXCLUDE list. Decrease number of modules being loaded, or move origin of self-configuration.
`No memory for io_init[], io_start[] or pwr_clr[].`	Unable to allocate memory for kernel data structures. Decrease number of modules being loaded, or move origin of self-configuration.
`No memory for parameter checking`	Unable to allocate memory for parameter checking. Decrease number of modules being loaded, or move origin of self-configuration.
`No memory for sys3bconfig structure`	Unable to allocate memory for sys3bconfig structure. Move origin of self-configuration.
`No memory for Xreloc`	Unable to allocate memory for relocation entry. Decrease number of modules being loaded, or move origin of self-configuration.
`No memory for Xsymbol`	Unable to allocate memory for internal symbol table. Decrease number of modules being loaded, or move origin of self-configuration.

Error Message	Description/Action
`Undefined expression element`	Expression element unknown. A master file has an invalid expression. See **master**(4) for valid expression element syntax. Check all master files for expression syntax.
`<driver>: character string initializer truncated`	This is a warning message. Attempting to initialize a string variable for <driver> but found string too long for variable. Variable initialized to truncated string. This may cause unusual side effects.
`<driver>: dependent driver <name> is EXCLUDED`	<Driver> has dependencies upon driver <name>, but driver <name> is marked to be excluded. <Driver> will not be loaded. Remove driver <name> from the **EXCLUDE** line of the system file, or add <driver> to the **EXCLUDE** line. If <driver> is added to **EXCLUDE** line, remove it from **INCLUDE** line if there.
`<driver>: dependent driver <name> not available`	<Driver> has dependencies upon driver <name>, but the object file for driver <name> is not found in boot directory. <Driver> will not be loaded. Place **mkboot**ed object file for driver <name> in **/boot** directory, or add <driver> to **EXCLUDE** line of system file. If <driver> is on **INCLUDE** line, remove it from that line.
`<driver>: device not equipped for dependent driver <name>`	<Driver> has dependencies upon driver <name>, but hardware not equipped for driver <name>. <Driver> will not be loaded. Either add hardware for driver <name>, or add <driver> to EXCLUDE line of the system file. If <driver> is added to the EXCLUDE line, then remove it from the INCLUDE if it exists there.
`<driver>: illegal character string initialization: zero assumed`	This is a warning message. Attempting to initialize a string variable for <driver> but found illegal character string. Variable initialized to zero. This may cause unusual side effects.
`<driver>: routine <name>: unknown id: RNULL assumed`	The routine <name> which is referenced by <driver> could not be found. It is resolved to rnull(), which may have unusual side effects.
`<file> does not exist`	Unable to find <file>.
`<file>: not object file and not ascii text file`	If <file> refers to the boot program, then this message means that <file> is not an object file. If <file> refers to a system file, that <file> was found to be non-ascii.

Error Message	Description/Action
`<name>: already allocated`	Self-configuration attempted to allocate space for variable <name> but found it was already allocated. The variable is from a master file variable list. This warning message may produce unusual side effects.
`<name>: already defined`	Self-configuration expects to define the symbol <name>, but found it already defined. This is a warning message, but could result in unusual side effects.
`<name>: data initializer #C (expression) unknown: zero assumed`	The master file for module <name> contains a reference to a master file entry "number of controllers" (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: data initializer #D (expression) unknown: zero assumed`	The master file for module <name> contains a reference to a master file entry "number of devices" (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: data initializer #M (expression) unknown: zero assumed`	The master file for module <name> contains a reference to a module major number (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: data initializer & expression cannot be resolved`	The master file for module <name> contains a reference to the address of a symbol (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: data initializer #expression unknown: zero assumed`	The master file for module <name> contains a reference to the size of a symbol (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: data initializer expression unknown: zero assumed`	The master file for module <name> contains a reference to a parameter (expression) which cannot be found. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: File too large`	The size of a file exceeded the maximum file size.

Error Message	Description/Action
`<name>: flagged as ONCE only: #C set to 1`	The master file for driver <name> is marked as only specify once, but the number of controllers is greater than one. The number of controllers is set to one. Correct master file <name>, mkboot driver <name>, and then reboot.
`<name>: I/O error`	Some physical input/output error has occurred while reading file <name>.
`<name>: invalid argument`	Some invalid argument was passed.
`<name>: invalid object file`	Object file <name> not valid for this machine.
`<name>: no drivers`	Unable to find valid loadable driver object files in boot directory <name>. Check path of boot directory, and that the boot directory contains mkbooted driver object files. Then reboot.
`<name>: no section headers`	The file header of boot module <name> indicates number of sections in object is zero. Recompile <name>, mkboot <name>, and then reboot.
`<name>: no such device`	An attempt was made to apply an inappropriate system call to a device; e.g., read a write-only device.
`<name>: no such file or directory`	The file <name> does not exist. This error occurs when a file is specified and the file should exist but doesn't, or when one of the directories in a path name does not exist.
`<name>: no symbols`	No symbols were found in the file specified for boot.
`<name>: not a directory`	<Name> is not a directory. A non-directory was specified where a directory is required, for example in a path prefix.
`<name>: not MAC32 magic`	Object module <name> contains an incorrect magic number. Recompile <name> with correct sgs, mkboot <name>, and then reboot.
`<name>: previously allocated`	Self-configuration attempted to allocate space for variable <name> but found it had been previously allocated by self-configuration. The variable is from a master file variable list. This warning message may produce unusual side effects.

Error Message	Description/Action
`<name>: previously defined`	Self-configuration expects to define the symbol <name>, but found it already defined. This is a warning message, but could result in unusual side effects.
`<name>: required driver is EXCLUDED`	The driver <name> is marked as being required in its master file, but is EXCLUDED in the system file. Unknown results may occur. Illegal to EXCLUDE a required driver. Remove <name> from the EXCLUDE line of the system file and reboot.
`<name>: routine <name>() not found`	The routine <name> was not found in the boot program <name>. The value of routine <name> is set to zero. While this is only a warning message, it may have unusual side effects.
`<name>: Special device cannot be used`	File <name> is a special device (character, block, or FIFO).
`<name>: truncated read`	Read of file <name> failed.
`<name>: truncated string table`	While reading string table of file <name>, end-of-file was encountered prematurely.
`Driver <name>: major number greater than 255`	A master file for software driver <name> contains a major number greater than 255. Correct major number in master file <name>. Then reboot.
`Driver <name>: missing section .text, .data or .bss`	Driver object module <name> is missing .text, .data, or .bss section header. Recompile driver <name>. Then reboot.
`Driver <name>: not a valid object file`	Driver <name> contains bad magic number. Recompile driver <name>. Then reboot.
`Driver <name>: not processed by mkboot(1M)`	Driver object file <name> was not processed by mkboot command. Run mkboot on driver file <name>. Then reboot.
`EXCLUDE: <name>: driver is INCLUDED`	Driver <name> to be excluded is also to be included. Remove <name> from one or the other in the system file. Then reboot.
`External symbol <name> is undefined: set to zero`	The external symbol <name> cannot be resolved. Its value is set to zero.

Error Message	Description/Action
`INCLUDE: <name>: device not equipped`	Hardware not equipped for driver <name> to be included. This is a warning and the driver <name> will not be loaded. Either add hardware for device <name>, add EXCLUDE statement to the system file for driver <name>, or remove driver from boot directory. Then reboot.
`INCLUDE: <name>: driver is EXCLUDED`	Driver <name> appears on both the INCLUDE and EXCLUDE lines of the system file. Remove <name> from one or the other in the system file. Then reboot.
`INCLUDE: <name>: driver not found`	Driver <name> is marked to be included, but unable to find its object file in the boot directory. If driver <name> is to be included, then run mkboot on <name> object file and reboot. If driver <name> was not to be included, then remove it from the INCLUDE line in the system file. Then reboot.
`Device <name> previously configured at board code <n>`	Device <name> has been moved. It was previously located in slot <n>. This is a warning message indicating a change in configuration was detected.
`Device <name> (board code <n>) not configured`	Device <name> located in slot <n> was not installed at the time the absolute boot image was created. Therefore, it will not be usable when this absolute boot image is used.
`Driver not found for <name> device (board code <n>)`	A driver for device <name> was not found in the boot directory. The device is located in slot <n>. This is a warning message. Add driver for device <name> and reboot.
`No memory for driver linked-list`	Unable to allocate memory to build the driver linked-list.
`No memory for kernel optional header`	Unable to allocate memory to build the kernel optional header.
`No memory for driver symbol table processing`	Unable to allocate memory to process a driver symbol table.
`No memory for symbol table`	Unable to allocate memory for kernel symbol table processing.

Error Message	Description/Action
`Section <name> (<file>) loaded below MAINSTORE address`	The section <name> from file <file> has an origin below the value of MAINSTORE.
`Section <name> (<file>) loaded beyond end of MAIN-STORE`	The section <name> from file <file> has an origin beyond the end of physical memory.
`Section <name> (<file>) over-laps boot program`	The section <name> from file <file> overlaid portions of self-configuration.
`Section <name> (<file>) over-laps <name> (<file>)`	The section <name> from file <file> overlays the section <name> from file <file>.
`VTOC does not exist or is damaged.`	Cannot find the volume table of contents on disk.
`VTOC read failed.`	Unable to read the volume table of contents on disk.
`Parameter <name> multiply defined`	A parameter <name> is found to be defined more than once with different values. If the conflict cannot be resolved, then the parameter value will be set to zero.
`<driver>: <name> = <n>`	Driver <driver> defines parameter <name> to be <n>.
`<driver>: <name> = <n> (<driver> EXCLUDED, parameter ignored)`	Driver <driver> defines parameter <name> to be <n>, but driver <driver> is to be EXCLUDED. Therefore, this definition will be ignored.
`<driver>: <name> = <n> (set to zero)`	Unable to resolve conflict of parameter <name>, its value is being set to zero. Driver <driver> defines parameter <name> to be <n>. Correct parameter definitions of the master files listed. Execute `mkboot` drivers for any changed master files, and then reboot.
`<driver>: <name> = <string>`	Driver <name> defines parameter <name> to be <string>.
`<driver>: <name> = <string> (<driver> EXCLUDED, parameter ignored)`	Driver <driver> defines parameter <name> to be <string>, but driver <driver: is to be EXCLUDED. Therefore, its definition will be ignored.
`<driver>: <name> = <string> (set to zero)`	Unable to resolve conflict of parameter <name>, its value is being set to zero. Driver <driver> defines parameter <name> to be <string>. Correct parameter definitions of the master files listed. Mkboot drivers for any master files which were changed, and then reboot.

The following error messages are generated by self-configuration when parsing the system file.

Error Message	Description/Action
`System: line #: cannot boot directory`	The file specified for booting is a directory.
`System: line #: cannot boot special device`	The file specified on line # of the system file, the file to boot, is a special device and cannot be loaded.
`System: line #: cannot boot special file`	The file specified on line # of the system file, the file to boot, is a special file and cannot be loaded.
`System: line #: count must be numeric`	The count value on system file line # is not a numeric value.
`System: line #: file not BLOCK or CHAR special`	The device on line # of the system file is not a block or character special device.
`System: line #: line too long`	Line # of the system file is longer than 256 characters.
`System: line #: major/minor must be numeric`	The major and/or minor numbers on line # of the system file are not numeric values.
`System: line #: must be numeric`	The numbers on line # are not numeric values.
`System: line #: no such file`	The file specified on line # of the system file, the file to boot, cannot be accessed.
`System: line #: path too long`	The path of the program to boot on system file line # contains more than 100 characters.
`System: line #: syntax error`	A syntax error was found on line # of the system file.

The following boot error and warning messages result from input to system file prompts by self-configuration.

Error Message	Description/Action
`System: cannot boot directory`	The file specified for booting is a directory.
`System: cannot boot: special device`	The file specified for boot is a special device file.
`System: cannot boot special file`	The file specified for boot is a special file.
`System: count must be numeric`	The count value is not numeric.
`System: file not BLOCK or CHAR special`	The device specified is not a character or block special device file.
`System: line too long`	The input line contains more than 256 characters.
`System: major/minor must be numeric`	The major and/or minor numbers are not numeric
`System: must be numeric`	The value entered is not numeric.
`System: no such file`	The file specified for boot cannot be accessed.
`System: path too long`	The path for the boot file contains more than 100 characters.
`System: syntax error`	The input line contains syntax errors.

Pump Error Messages

pump is a user-level command that downloads firmware to feature cards mounted in the 3B2 Computer backplane slots during the power-up sequence. pump error messages appear on the console terminal when a phase in the auto-pump sequence fails. Although these errors are not fatal to the entire system, the affected card is not operational. Therefore, normal services provided by the device are not accessible.

Error Message (See NOTEs below)	Description/Action
Pump: */dev/devname* **returned a CIO FAULT during** *phase*	Turn power off using power switch. After powerdown sequence has completed, turn power on again. If error occurs again, contact your service representative. The UNIX system may panic soon.
Pump: */dev/devname* **returned a CIO invalid Queue Entry during** *phase*	Same action as above.
Pump: */dev/devname* **did not respond during** *phase*	Same action as above.
Pump: */dev/devname* **did not respond during** *phase*	Same action as above.
Pump: A timeout has occurred on */dev/devname* **during** *phase*	Same action as above.
Pump: There was no return for */dev/devname* **during** *phase*	Same action as above.
Pump Error: 208-ioctl call	

NOTE

The term */dev/devname* refers to `/dev/ttyAB`, `/dev/NI`, and so on, where:

A = Feature Card slot on backplane
B = Port on Expansion Port Feature Card
NI = Network Interface Feature Card

NOTE

The term *phase* refers to one of the following phases:

`Reset` = Reset of the Feature Card so that pumping can occur
`Download` = Pumping to firmware of Feature Card
`Sysgen` = Initialization of Feature Card to known state
`Force call to function` = Calling the starting address of firmware that was downloaded.

Basic Networking Utilities Error Messages

This section lists the error messages associated with the Basic Networking Utilities. There are two types of error messages. `ASSERT` errors are recorded in the `/var/uucp/.Admin/errors` file. `STATUS` errors are recorded in individual machine files found in the `/var/uucp/.Status` directory.

BNU ASSERT Error Messages

When a process is aborted, `ASSERT` error messages are recorded in `/var/uucp/.Admin/errors`. These messages include the file name, SCCS ID, line number, and text listed below. In most cases, these errors are the result of file system problems. The `errno` (when present) should be used to investigate the problem. If `errno` is present in a message, it is shown as () in the following list.

Error Message	Description/Action
`CAN'T OPEN`	An open() or fopen() failed.
`CAN'T WRITE`	A write(), fwrite(), fprint(), etc. failed.
`CAN'T READ`	A read(), fgets(), etc. failed.
`CAN'T CREATE`	A create() call failed.
`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.
`CAN'T CHMOD`	A chmod() call failed.
`CAN'T LINK`	A link() call failed.
`CAN'T CHDIR`	A chdir() call failed.
`CAN'T UNLINK`	A 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 `/var/uucp/.Corrupt` directory failed. The directory is probably missing or has wrong modes or owner.

Error Message	Description/Action
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 file system is in trouble, or the /etc/passwd file is inconsistent.
BAD LOGIN_UID	Same as previous.
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/strange request was attempted. Contact your AT&T Account Representative or Authorized Dealer.
TOO MANY SAVED C FILES	Same as previous.
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/Systems files (Class field).
PERMISSIONS file: BAD OPTION	There is a bad line or option in the Permissions file. Fix it immediately!
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.

Error Message	Description/Action
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! Contact your AT&T Account Representative or Authorized Dealer.
XMV ERROR	There is a problem with some file or directory. It is likely the spool directory, since 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 will be attempted later (uuxqt). No action need be taken.

BNU STATUS Error Messages

Status error messages are messages that are stored in the /var/uucp/.Status directory. This directory contains a separate file for each remote machine that your 3B2 Computer attempts to communicate with. These individual machine files contain status information on the attempted communication, whether it was successful or not. What follows is a list of the most common error messages that may appear in these files.

Error Message	Description/Action
OK	Things are OK.
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.

Error Message	Description/Action
LOGIN FAILED	The login for the given machine failed. It could be a wrong login/password, wrong number, a very slow machine, or failure in getting through the Dialer-Token-Pairs 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/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 **/var/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.
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 3B2 Computer.
REMOTE HAS A LCK FILE FOR ME	The remote site has a LCK file for your 3B2 Computer. They could be trying to call your machine. If they have an older version of Basic Networking, the process that was talking to your machine may have failed leaving the LCK file. If they have the new version of Basic Networking, and they

Error Message	Description/Action
	are not communicating with your 3B2 Computer, 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 3B2 Computer in its **Systems** file.
`REMOTE REJECT AFTER LOGIN`	The login used by your 3B2 Computer to login does not agree with what the remote machine was expecting.
`REMOTE REJECT, UNKNOWN MES-SAGE`	The remote machine rejected the communication with your 3B2 Computer for an unknown reason. The remote machine may not be running a standard version of Basic Networking.
`STARTUP FAILED`	Login succeeded, but initial handshake failed.
`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.

MAIL SUBSYSTEM ADMINISTRATION

F Mail Subsystem Administration

Administering the Mail Subsystem

The purpose of this appendix is to help a system administrator take advantage of various options within the mail subsystem. By default, the mail subsystem provides electronic communications between users on the same machine, or between machines connected together on a UUCP network, and supports two addressing schemes, known as "bang" style and "domain" style. The system administrator does not need to do anything for mail to work in the default manner.

This appendix will provide information on setting up a smarter host, establishing a domain name, administering a set of sites so that they all send mail with the same machine name, setting up the mail directory to be shared across a networked file system, such as RFS and NFS, filling in alias information, and setting up a connection to another site that uses the Simple Mail Transfer Protocol (SMTP).

Mail Administration Files

There are four files that are important to mail administration. The surrogate file, `/etc/mail/mailsurr`, is described on the `mailsurr`(4) manual page. It describes how to rewrite addresses and how to deliver messages through the networks. The configuration file, `/etc/mail/mailcnfg`, is described on `mailcnfg`(4). It permits various per-site options to be established. For more information on the `mailsurr` and `mailcnfg` manual pages, see the *System Administrator's Reference Manual*. The master alias path file, `/etc/mail/namefiles`, points to one alias file, `/etc/mail/names`. Both are described on `mailalias`(1). These two files are used to define name mappings and address lists. For more information on the `mailalias` manual page, see the *User's Reference Manual*.

Mail Addressing Styles

The default surrogate file contains entries to translate between domain style addresses and bang style addresses. Bang style addressing is characterized by exclamation points (a.k.a. bangs) within the address and looks like *host*`!`*user* or *host1*`!`*host2*`!`*user*. Domain style addressing is characterized by the commercial at sign (`@`) and looks like *user*`@`*host*`.`*domain* or *user*`@`*host*.

Establishing a Smarter Host

Although it is possible to maintain the data files for the UUCP network so that the system knows about hundreds or thousands of other systems that can be contacted, it is impractical to do so. It is often much easier to set up what is known as a "smarter host," that is, another UNIX system to which remote mail will be shipped if the local machine doesn't know about the system to which the mail is being sent. For example, assume you need to send a mail message to `hosta!tony`, but your local machine does not know about `hosta`. The mail message can be automatically routed to the machine `worldly`, which has a more extensive list of UUCP connections.

This is done in two steps:

1. Add a line to `mailcnfg` that says

   SMARTERHOST=*smhost*

   where *smhost* is replaced with the name of the smarter system (`worldly`).
2. Remove the # character from the line within `mailsurr` that looks like this:

```
#'.+'  '.*[!@].*'    'Translate R=%X!%n'
```

Establishing Domain Addresses

As distributed, mail knows about two forms of domain style addresses:

user@host

and

user@host.UUCP

It does not know about

user@host.domain

A domain name is an internationally recognized and registered name for a set of machines. Commercial entities may be registered under domain names similar to .*company-name*.COM and educational entities may be registered under domain names similar to .*school*.EDU. (Note that .UUCP is not a true domain name. The high-level domain names of .COM and .EDU are assigned by a central authority.) A system will generally know how to establish direct connections to other machines within the local domain, but will want to make use of a smarter host to take care of other domains.

To establish the local domain name, type

```
        /usr/sbin/domainname domain
```

where *domain* is replaced with the domain name, such as *.company-name*.COM or whatever is appropriate, and contains a leading period. (Any periods in the domain name will be converted to `\.` before being passed to the regular expressions in the surrogate file.)

The domain name will also be used by the SMTP router when rewriting header files into RFC822 format (see "Administering SMTP" below).

Establishing a Mail Cluster or Gateway

With the arrival of inexpensive personal computers, it is often desirable to assign a single name to a set (or a cluster) of machines by which all the machines in the cluster will be known to external machines, for purposes of mail. For example, a cluster of machines known internally under names such as `Xsysa`, `Xsysb` and `Xsysc`, could be assigned the cluster name of `Xsys`. Mail sent from any of these machines would be shown as being from `Xsys`; that is, the internal names would not be known outside the cluster.

To establish a cluster name, add a line to `mailcnfg` that says:

```
        CLUSTER=extname
```

where *extname* is the name by which the machine is known externally (`Xsys`).

Establishing Mail Service on a Networked File System (RFS or NFS)

With the arrival of inexpensive Local Area Networking (LAN) and networked file systems such as RFS and NFS, clusters of machines that share many file systems can be set up. It is also possible to share `/var/mail` across the machines. In this case, you can arrange to have all users' mailboxes created on only one machine, but accessible from all machines.

As an example, assume that you want the machines Xsysa, Xsysb and Xsysc to share the mail directory under Xsysa. In addition, the entire file system for each system is mounted under the names /Xsysa, /Xsysb and /Xsysc. All users have home directories under file systems named /homea, /homeb and /homec, which are mounted on the corresponding machines.

To establish a shared /var/mail file system, complete the following steps:

1. Make certain that /var/mail from Xsysa is advertised.
2. Remove the directory /var/mail/:saved from the systems that will not have a local /var/mail (Xsysb and Xsysc).
3. Add a line to mailcnfg that says

   ```
   FAILSAFE=Xsysa
   ```

   With this specified, mail will look for the presence of /var/mail/:saved. If the directory is not there (indicating that the network connection to Xsysa has been lost), mail will requeue the file to be delivered to Xsysa via other means (such as UUCP or SMTP).
4. Move any mailboxes from /var/mail on Xsysb and Xsysc to Xsysa (otherwise the files will be inaccessible).
5. Mount /var/mail from Xsysa.
6. To allow the notify program to identify where the user is logged in (so that it can notify the user when new mail arrives), create a file on all machines named /etc/mail/notify.sys with contents similar to the following:

   ```
   Xsysa    /Xsysa
   Xsysb    /Xsysb
   Xsysc    /Xsysc
   ```

   The first column lists the name of the system and the second gives a pathname of the root file system for each machine.
7. To allow the notify program to handle a network failure, create a file on all machines named /etc/mail/notify.fsys with contents similar to the following:

```
/homea    Xsysa
/homeb    Xsysb
/homec    Xsysc
```

The first column lists a file system name and the second column contains the system (machine name) on which that file system is normally mounted. If `notify` cannot open the mail file for writing, it will look up the file system in this list and requeue the file to be delivered to the corresponding system via other means (such as UUCP or SMTP).

Administering `alias` Lists

Before delivering a local mail message, `mail` will look up the user name to see if it has been aliased to another name or list of names. The master alias path file `/etc/mail/namefiles` contains a list of files that `mail` will search for aliases. As distributed, this list contains only one file, `/etc/mail/names`, to be searched for aliases. If the named alias is found at the beginning of a line within an alias file, the rest of that line will be used as the alias. This may contain a single name, or a list of names separated by whitespace. For example, if you want to set up a group mailing list (such as `andy.group`) that will be expanded, add a line similar to the following to the alias file:

```
andy.group tony paul john ned gary hailey mike
```

Recursive references are permitted, as in this reference to `andy.group` within another alias:

```
armida.dept andy.group danielle.group bob.group \
        lee.group pier.group
```

Several alias files can be listed in `namefiles`, which may be kept anywhere on the machine. This permits different alias files to be owned by different administrators.

Other Tricks of the `surrogate` File Trade

Logging Mail

Occasionally it may be necessary to keep a log of traffic going through the system. For example, if you were to write a program called `/usr/lib/mail/surrcmd/logmail` that takes three arguments (a log file name, the sender and recipient), it could log all external mail flowing through the system by using this surrogate entry:

```
'.+!.+' '.*' '> /usr/lib/mail/surrcmd/logmail /var/adm/mailtransport %R %n'
```

Another example would be to log traffic to or from a particular system (here to `xyz` and from `abc`):

```
'.*' 'xyz!.+' '> /usr/lib/mail/surrcmd/logmail /var/adm/mailto-xyz %R %n'
'abc!.+' '.*' '> /usr/lib/mail/surrcmd/logmail /var/adm/mailfrom-abc %R %n'
```

Path Translation

Many systems have a path translation program available that will give the shortest route to a given system, based on various criteria or a database. An example of this is the public domain `smail` program. As an alternative to using a smarter host, the autorouter can be invoked as a final step in the `mailsurr` file:

```
'.+'      '.*[!@].*'                  'Translate R=|smail -A %n'
```

Controlling Mail Resource Access

It is often necessary to control access to commercial services, such as AT&T Mail. One method of doing this is to prevent any non-local users from sending mail to the commercial site using the `Accept` and `Deny` commands:

```
'[^!]+' 'attmail!.+' 'Accept'
'.+'    'attmail!.+' 'Deny'
```

Another method is to use an external program to check the sender's path to see if it is a valid user of the service. For example, this shell script returns 0 if the sender is a valid system, and 1 otherwise:

```
case "$1" in
    abc | def | ghi ) exit 0 ;;
    * ) echo "$1 is not permitted to send mail to external service"
         exit 1 ;;
esac
```

If the script were installed as `/usr/lib/mail/surrcmd/chksender`, it would be invoked as a delivery agent that will either continue or fail:

```
# check senders more than one hop away
'.+!(.+)![^!]+' 'attmail!.+' '< C=0;F=*; /usr/lib/mail/surrcmd/chksender \\1'
# check senders one hop away
'(.+)![^!]+'     'attmail!.+' '< C=0;F=*; /usr/lib/mail/surrcmd/chksender \\1'
```

Administering SMTP

The Simple Mail Transfer Protocol (SMTP) mail subsystem is delivered as a group of programs that allow UNIX system mail to send and receive mail using the SMTP protocol. This protocol is typically used over TCP/IP networks. However, as delivered, the SMTP processes can connect over any TLI-based, connection-oriented, transport that has been administered to have an SMTP service.

To establish SMTP service requires these steps:

1. By default, SMTP is installed in the mail surrogate file, but it is turned off (commented out). It may be turned on by uncommenting the line that says:

   ```
   #'.+' '([^!@]+)!(.+)' '< /usr/lib/mail/surrcmd/smtpqer %R \\1 \\2'
   ```

   To uncomment this line, edit the `mailsurr` file to remove the # from the start of each line.

   Mail is addressed using the standard UNIX system mail formats of *host*!*user* or *user*@*host*. If *host* is known to support SMTP mail delivery, the mail will be queued for delivery using SMTP. If not, `smtpqer` will not accept the message, and delivery will be done by subsequent surrogates in the `mailsurr` file.

   All messages that are spooled for SMTP delivery are stored in the directory `/var/spool/smtpq/`*host*, where *host* is the name of the machine to which mail is being sent.

2. The list of machines that will accept SMTP mail is specified by the `netdird` service. See `netdird`(1M) in the *System Administrator's Reference Manual* to see how to add services to this database. By default, the SMTP daemon `smtpd` will always start when your system is booted. If `smtpd` finds that there are no networks installed for which the SMTP service is defined, it will exit.

When the daemon smtpd receives a piece of mail, it does three things: (1) it inserts a valid UNIX system mail "From " header line; (2) it converts the recipient address to *host*!*user* form; and, (3) it hands the message to rmail for delivery.

3. The following entry must be uncommented from root's crontab file (see crontab(1) for an explanation of this file):

```
25 * * * * /usr/lib/mail/surrcmd/smtpsched
55 1 * * * /usr/lib/mail/surrcmd/smtpsched -c -w 1 -r 7
```

To do this, execute the following commands as root:

```
# crontab -l > /tmp/cron.temp
# ed /tmp/cron.temp
g!/smtpsched!s/^#//
w
q
# crontab /tmp/cron.temp
# rm /tmp/cron.temp
#
```

By default, mail that cannot be delivered immediately (as it is sent) is queued and retried at one hour intervals by smtpsched. You can change the interval by modifying the entry for smtpsched in root's cron file.

SMTP logs all SMTP activity, including incoming mail messages, in the log file /var/spool/smtpq/LOG. It is backed up once per day by smtpsched; previous days' log files are located in /var/spool/smtpq/LOG.*n*, where *n* is the day of the week (from 0 to 6). The smtpsched program will also return undeliverable mail messages. For more information on smtpsched(1M) see the *System Administrator's Reference Manual.*

Setting Up SMTP to Listen Over Multiple Networks

smtpd will listen to any connection-oriented TLI network that provides the SMTP service. TLI networks are specified in /etc/netconfig. For each network that is connection-oriented, smtpd will use netdir_getbyname(3) to determine if the SMTP service exists for that network. If the service does exist, a port is opened at the address returned by this function. To make the listener listen to a new network, first administer the netdir databases, and then restart the listener.

Glossary

address	a number, label, or name that indicates the location of information in the computer's *memory.*
advertise	a means of making *resources* available from a local *host* to other *hosts* in a *Remote File Sharing* environment.
a.out	the default name of a freshly compiled *object file,* pronounced 'A-dot-out'; historically a.out signified assembler output.
archive	1. a collection of data gathered from several *files* into one file. 2. especially, such a collection gathered by **ar**(1) for use as a library.
automatic calling unit	a hardware *device* used to dial stored telephone numbers; allows the system to contact another system over phone lines without manual intervention.
bad block	a section of a storage medium which cannot store data reliably.
block	the basic unit of *buffering* in the *kernel,* 1024 bytes; see *indirect, logical,* and *physical blocks.*
block device	a *device* upon which a *file system* [1] can be *mounted,* typically a permanent storage device such as a tape or disk drive, so called because data transfers to the device occur by *blocks;* cf. *character device.*
boot	to start the operating system, so called because the *kernel* must bootstrap itself from secondary storage into an empty machine. No *login* [3] or *process* persists across a boot. **boot block** the first block of a *file system* [1], which is reserved for a booting program.
boot program	loads the *operating system* into *core.*
buffer	1. a staging area for input-output where arbitrary-length transactions are collected into convenient units for system operations; the *file system* [3] uses buffers, as does *stdio.* 2. to use buffers.

buffer pool	a region of store available to the *file system* [3] for holding *blocks;* all but *raw* [2] input-output for *block devices* goes through the buffer pool so read and write operations may be independent of device blocks.
cartridge tape	a storage medium that consists of a magnetic tape wound on spools housed in a plastic container.
character device	a *device* upon which a *file system* [1] cannot be *mounted* such as a terminal or the *null device.*
child process	see *fork.*
client	a *host* that has *mount*ed an *advertise*d *resource* from another *host* in a *Remote File Sharing* environment.
command	1. an instruction to the *shell,* usually to run a *program* [1] as a *child process.* 2. by extension, any *executable file,* especially a *utility program.*
command file	same as *shell script.*
configuration	the arrangement of the software or hardware of a system, peripheral, or network as defined by the nature, number, and chief characteristics of its functional units.
controller	a *device* that directs the transmission of data over the data links of a *network.*
core file	a *core image* of a terminated *process* saved for debugging; a core file is created under the name 'core' in the *current directory* of the process.
core image	a copy of all the *segments* of a running or terminated program; the copy may exist in main store, in the *swap area,* or in a *core file.*
crash	If a hardware or software *error* condition develops that the system can't handle, it takes itself out of service, or crashes. Such conditions occur when the system can't allocate resources, manage *processes,* respond to requests for system functions, or when the electrical power is unstable.

cron	a command which creates a daemon that invokes commands at specified dates and times.
cylinder	the set of all *tracks* on a *disk* which are the same distance from the axis about which the disk rotates.
daemon	a *background* process, often perpetual, that performs a system-wide public function, e.g. **calendar** (1) and **cron** (8); the affected spelling is an ancient legacy.
destination	the remote system that will ultimately receive a *file* transferred over a *network*.
device	1. a *file* [2] that is not a *plain file* or a *directory*, such as a tape drive, or the *null device;* a *special file*. 2. a physical input-output unit.
diagnostic	a message printed at your terminal that identifies and isolates *program* errors.
directory	a *file* that comprises a catalog of *filenames* [2]; the organizing principle of the *file system* [2], a directory consists of *entries* which specify further *files* (sense 2, including directories), and constitutes a node of the *directory tree*.
directory entry, entry	1. an association of a name with an *inode number* appearing as an element of a *directory*. 2. the name part of such an association.
directory hierarchy	the tree of all *directories*, in which each is reachable from the *root* via a chain of *subdirectories*.
directory tree	same as *directory hierarchy*.
disk	a platter coated with magnetic material on which data can be stored.
diskette	a magnetic storage medium which is smaller and more flexible than a hard *disk*.
domain	a logical grouping of *hosts* in a *Remote File Sharing* environment. Each *host* in a *domain* relies on the same *domain name server*(s) for certain *resource* sharing and security services. Each *domain* has one

	primary and zero or more *secondary domain name servers.*
domain name server	a computer that creates and maintains the following information for *hosts* in a *Remote File Sharing* domain: *advertised resources, host* names and passwords, names and addresses for name servers of other domains (optional), *host* user and group information used for *ID mapping* (optional).
drive	the hardware device that holds magnetic disks, diskettes, and tapes while they are in use.
dump	a copy of the *core image* of the operating system.
environment	1. a set of strings, distinct from the *arguments,* made available to a *process* when it *executes* [2] a *file;* the environment is usually inherited across *exec*(2) operations. 2. a specific environment [2] maintained by the *shell.* 3. a nebulously identified way of doing things, as in 'interactive environment': a deprecated usage, not always expunged from these manuals.
error	occurs when a hardware or software condition prevents the successful *execution* of a system or a user *process.*
error message	a message sent from the system to the *system console* when an *error* occurs.
exec	a system call which allows the user to request the execution of another program.
executable file	1. an *object file* that is ready to be copied into the *address* space of a *process* to run as the code of that process. 2. a file that has execute *permission,* either an *executable file* [1] or a *shell script.*
execute	1. informally, to run a *program.* 2. to replace the *text segment* and *data segments* of a *process* with a given *program* [1].

FIFO	a named permanent *pipe* which allows two unrelated *processes* to exchange information using a pipe connection.
file	1. in general, a potential source of input or destination for output. 2. most specifically, an *inode* and/or associated contents, i.e. a *plain file*, a *special file*, or a *directory*. 3. a *directory entry*; several directory entries may name the same file [2]. 4. most loosely, a *plain file*.
file descriptor	a conventional integer quantity that designates an *open file*.
filename	1. a *pathname*. 2. the last component name in a pathname.
file system	1. a collection of *files* that can be *mounted* on a block *special file*; each file of a file system appears exactly once in the *i-list* of the file system and is accessible via some *path* from the *root* directory of the file system. 2. the collection of all *files* on a computer. 3. the part of the kernel that deals with file systems [1].
filter	a *program* [1] that reads from the *standard input* and writes on the *standard output*, so called because it can be used as a data-transformer in a *pipeline*.
firmware	the Non-Volatile Random Access Memory (NVRAM), which permanently holds a few, special programs.
floppy key	a copy of the default *firmware* password for a 3B2 computer on a *diskette*. It may be used to reset the password to its original value.
flush	to empty a *buffer*, for example to throw away unwanted input-output upon *interrupt* or to release output from the clutches of *stdio*.

fork	to split one *process* into two, the **parent process** and **child process**, with separate, but initially identical, *text, data,* and *stack segments.*
formatting	the process of imposing an addressing scheme on a *disk.* This includes the establishment of a *VTOC,* and the mapping of both sides of the disk into *tracks* and *sectors.*
free list	in a *file system* [1], the list of *blocks* that are not occupied by data.
getty	one of a series of *processes* which connect the user to the UNIX system. *getty* is invoked by *init,* and in turn invokes *login.*
group	1. a set of *permissions* alternative to *owner* permissions for access to a *file.* 2. a set of *userids* that may assume the privileges of a group [1]. 3. the *groupid* of a file.
groupid	an integer value, usually associated with one or more *login names;* as the *userid* of a process becomes the *owner* of files *created* by the process, so the groupid of a process becomes the *group* [3] of such files.
hole	a gap in a *plain file* caused by *seeking* while writing; *read*(2) takes data in holes to be zero; a *block* in a hole occupies no space in its *file system.*
host	a computer that is configured to share *resources* in a *Remote File Sharing* environment.
ID mapping	a means of setting the permissions that each remote user and group will have for a *host's advertised resources* in a *Remote File Sharing* environment.
i-list	the index to a *file system* [1] listing all the *inodes* of the file system; cf. *inode number.*
indirect blocks	data blocks that are not directly referenced by a *inode* (because the file is larger than 10 1024-byte blocks); the inode has 3 *addresses* that indirectly reference (by a cascade of pointers) some 2,114,114

	data blocks (an extremely large potential *file* size). the inode has 1 address that points to 128 more data blocks; a second address that points to 128 blocks that each point to 128 data blocks; and finally a third address that points to 128 blocks each of which point to another 128 blocks, each of which point to 128 data blocks!
init	a general *process* spawner which is invoked as the last step in the *boot* procedure; it regularly checks a table that defines what processes should run at what *run level.*
inode	an element of a *file system* [1]; an inode specifies all properties of a particular *file* [2] and locates the file's contents, if any.
inode number, i-number	the position of an *inode* in the *i-list* of a *file system* [1].
instruction	see *address.*
integrity	in a *file system,* the quality of being without errors due to *bad blocks.*
interface programs	*shell scripts* furnished with the LP *spooling* software which interface between the user and the printer.
interrupt	1. a *signal* that normally terminates a *process,* caused by a break or an interrupt character. 2. a signal generated by a hardware condition or a peripheral *device.* 3. loosely, any *signal.*
IPC	an acronym for interprocess communication.
kernel	the UNIX system proper; resident code that implements the *system calls.*
kernel address space	a portion of memory used for data and code addressable only by the *kernel.*
line discipline	a module to handle protocol or data conversion for a *stream* [2]. A line discipline, unlike a *filter,* is part of the *kernel.*

link	1. to add an entry for an existing *file* to a directory; converse of *unlink*. 2. by extension, a *directory entry*. 3. loosely, any but one putatively primary directory entry for a given *inode*; either linked [1] or a *symbolic link*.
link count	the number of *directory entries* that pertain to an *inode*; a *file* ceases to exist when its link count becomes zero and it is not *open*.
load device	designates the physical *device* from which a program will be loaded into main *memory*.
log files	contain records of transactions that occur on the system; software that *spools*, for example, generates various log files.
logical block	a unit of data as it is handled by the software; the UNIX system handles data in 1024-byte logical blocks.
login	1. the *program* that controls logging in. 2. the act of *logging in*. 3. by extension, the computing session that follows a login [2].
memory	1. same as *memory image*. 2. physical memory represents the available space in main memory; *programs* are either *swapped* or *paged* into physical memory for *execution*. 3. virtual memory management techniques permit *programs* to treat *disk* storage as an extension of main memory.
memory image	same as *core image*.
mode, file mode	the *permissions* of a *file*; colloquially referred to by a 3-digit octal number, e.g. 'a 755 file'; see *chmod*(1).
mount	to extend the *directory hierarchy* by associating the *root* of a *file system* [1] with a *directory entry* in an already mounted file system; converse is unmount, spelled 'umount'.

namelist	same as *symbol table.*
network	the hardware and software that constitute the interconnections between computer systems, permitting electronic communication between the systems and associated peripherals.
networking	for computer systems, means sending data from one system to another over some communications medium (coaxial cable, phone lines, etc.). Common networking services include *file* transfer, remote *login*, remote *execution.*
node name	an up-to-six character name for the system; used as the official name of the machine in a *network.* The node name resides in the NODE parameter.
null device	a *device* [1] that always yields *end of file* on reading and discards all data on writing.
object file	a *file* of machine language code and data; object files are produced from source programs by compilers and from other object files and libraries by the link editor; an object file that is ready to run is an *executable file* [1].
operating system	the *program* for managing the resources of the computer. It takes care of such things as input/output procedures, process scheduling, the file system, removing this burden from user programs.
open file	1. the destination for input or output obtained by *opening* a *file* or creating a *pipe;* a *file descriptor;* open files are shared across *forks* and persist across *executes* [2]. 2. loosely, a file that has been opened, however an open file [1] need not exist in a *file system* [1], and a file [2] may be the destination of several *open files* simultaneously.
other	1. a set of *permissions* regulating access to a *file* by processes with *userid* different from the *owner* and *groupid* different from the *group* of the file. 2. the customary name of the default *group* [2] assigned upon *login.*

owner	the *userid* of the *process* that created a *file;* the owner has distinctive *permissions* for a file.
page	a fixed length, 1024-byte block that has a virtual *address,* and that can be transferred between main and secondary storage.
paging	the process by which *programs* are truncated into *pages* and transferred between main and secondary storage by the virtual handler (or paging *daemon*).
parent process	see *fork.*
partitions	units of storage space on disk.
path, pathname	a chain of names designating a *file;* a **relative pathname** leads from the current directory, for example, a path to *directory* A, thence to directory B, thence to *file* C is denoted A/B/C; a **full pathname** begins at the *root,* indicated by an initial '/', as in /A/B/C.
permission	a right to access a *file* in a particular way; read, write, execute (or look up in, if a directory); permissions are granted separately to *owner, group,* and *others.* **permission bit** a permission, so called because each permission is encoded into one bit in an *inode.*
physical block	a unit of data as it is actually stored and manipulated; the UNIX system handles data in 1024-byte physical blocks.
physical	see *memory.*
pipe	a direct stream connection between *processes,* whereby data written on an *open file* in one process becomes available for reading in another.
pipeline	a sequence of *programs* [1] connected by *pipes.*
polling	the interrogation of *devices* by the *operating system* to avoid contention, determine operation status, or ascertain readiness to send or receive data.

ports	the point of physical connection between a peripheral *device* (such as a terminal or a printer) and the device *controller* (ports board), which is part of the computer hardware.
primary name server	the computer on which administration for a *Remote File Sharing domain* is performed.
process	a connected sequence of computation; a process is characterized by a *core image* with instruction location counter, *current directory,* a set of *open files, control terminal, userid,* and *groupid.*
process id	an integer that identifies a *process.*
process number	same as *process id.*
profile	1. an optional *shell script,* '.profile', conventionally used by the *shell* upon *logging in* to establish the *environment* [3] and other working conditions customary to a particular user. 2. to collect a histogram of values of the instruction location counter of a *process.*
program	1. an *executable file.* 2. a *process.* 3. all the usual meanings.
queue	a line or list formed by items in a system waiting for service.
raw device	a *block device,* read and write operations to which are not *buffered,* and are synchronized to natural records of the physical *device.*
reboot	same as *boot.*
region	a group of machine *addresses* that refer to a base address.
release	a distribution of fixes or new functions for an existing software product.
Remote File Sharing	a software utilities package that enables computers to share *resources* across a network.

resource	a directory that is *advertised* in a *Remote File Sharing* environment. When a *resource* is *mount*ed on a *client*, the contents of the directory (files, devices, and named pipes) and any of its subdirectories are potentially available to users on the *client*.
re-tension	the process of re-winding the tape in a *cartridge tape device* to make sure it is at the correct tautness for accurate recording of data.
root	1. a distinguished directory that constitutes the origin of the *directory hierarchy* in a *file system* [1]. 2. specifically, the origin for the *file system* [2], with the conventional *pathname* '/'. 3. the origin of the directory hierarchy in a *file system* [1].
rotational gap	the gap between the actual *disk* locations of blocks of data belonging to the same *file*; the rotational gap compensates for the continuous, high-speed rotation of the disk so that when the controller is ready to reference the next physical block the read-write head is positioned correctly at the beginning of that block.
run level	a software *configuration* of the system which allows a particular group of *processes* to exist.
schedule	to assign resources— main store and CPU time—to *processes*.
scheduler	a permanent *process*, with *process number* 1, and associated *kernel* facilities that does scheduling.
search path	in the *shell*, a list of *pathnames* of *directories* that determines the meaning of a *command*; the command name is prefixed with members of the search path in turn until a pathname of an *executable file* [2] results; the search path is given by the shell variable PATH.
secondary name server	a *host* that is configured to take over *domain name server* responsibilities temporarily in case the *primary name server* goes down.

section, sector	A 512-byte portion of a *track* which can be accessed by the magnetic disk heads in the course of a predetermined *rotational* displacement of the storage device.
segment	a contiguous range of the address space of a *process* with consistent store access capabilities; the four segments are (i) the **text segment**, occupied by executable code, (ii) the **data segment**, occupied by *static* data that is specifically initialized, (iii) the **bss segment**, occupied by static data that is initialed by default to zero values, and (iv) the **stack segment**, occupied by *automatic* data, see *stack*; sometimes (ii), (iii), and (iv) are collectively called data segments.
semaphore	an IPC facility which allows two or more processes to be synchronized.
server	a *host* that is actively sharing one of its *advertised resources* with another *host* in a *Remote File Sharing* environment.
set userid	a special *permission* for an *executable file* [1] that causes a *process* executing it to have the access rights of the *owner* of the file; the owner's *userid* becomes the **effective userid** of the process, distinguished from the **real userid** under which the process began.
set userid bit	the associated *permission bit.*
shared memory	an IPC facility which allows two or more processes to share the same data space.
shell	1. the program *sh*(1), which causes other programs to be executed on *command;* the shell is usually started on a user's behalf when the user *logs in.* 2. by analogy, any program started upon logging in.
shell script	an executable *file* of *commands* taken as input to the *shell.*

signal	an exceptional occurrence that causes a *process* to terminate or divert from the normal flow of control; see *interrupt, trap.*
single-user	a state of the operating system in which only one user is supported.
source file	1. the uncompiled version of a *program*. 2. generally, the unprocessed version of a *file.*
special file	an *inode* that designates a *device,* further categorized as either (i) a **block special file** describing a *block device,* or (ii) a **character special file** describing a *character device.*
spool	to collect and serialize output from multiple *processes* competing for a single output service.
spool area	a *directory* in which a spooler collects work.
spooler	a *daemon* that spools.
stack	a *segment* of the *address* space into which *automatic* data and subroutine linkage information is allocated in last-in-first-out fashion; the stack occupies the largest data addresses and grows downward towards *static* data.
standard error	one of three files described below under *standard output.*
standard input	the second of three files described below under *standard output.*
standard output	*open files,* customarily available when a *process* begins, with *file descriptors* 0, 1, 2 and *stdio* names 'stdin', 'stdout', 'stderr'; where possible, utilities by default read from the standard input, write on the standard output, and place error comments on the standard error file. Initially, all three of these files default to your terminal.

startup	same as *boot*
sticky bit	a *permission* flag that identifies a file as a *sticky file.*
sticky file	a special *permission* for a *shared text* file that causes a copy of the *text segment* to be retained in the *swap area* to improve system response.
super block	the second *block* in a *file system* [1], which describes the allocation of space in the file system; cf. *boot block.*
	userid 0, which can access any *file* regardless of *permissions* and can perform certain privileged *system calls,* e.g. setting the clock.
swap	to move the *core image* of an executing program between main and secondary storage to make room for other *processes.*
swap area	the part of secondary store to which *core images* are *swapped;* the swap area is disjointed from the *file system.*
symbolic link	an *inode* that contains the *pathname* of another. References to the symbolic link become references to the named inode.
symbol table	information in an *object file* about the names of data and functions in that file; the symbol table and *address* relocation information are used by the link editor to compile *object files* and by debuggers.
System Administration	when capitalized, refers to the package of screens and interactive prompts, invoked through the sysadm(1) command, that help you accomplish most system administration tasks.
system calls	1. the set of system primitive functions through which all system operations are allocated, initiated, monitored, manipulated, and terminated. 2. the system primitives invoked by user *processes* for system-dependent functions, such as I/O, process creation, etc.

system console	the directly connected terminal used for communication between the operator and the computer.
system name	an up-to-six character name for the system; resides in the SYS parameter.
table	an array of data each item of which may be uniquely identified by means of one or more arguments.
text file,	a *file,* the bytes of which are understood to be in ASCII code.
track	an addressable ring of *sections* on a *disk* or *diskette;* each disk or diskette has a predefined number of concentric tracks, which allows the disk head to properly access *sections* of data.
trap	a method of detecting and interpreting certain hardware and software conditions via software; a trap is set to catch a *signal* (or *interrupt*), and determine what course of action to take.
tunable parameters	variables used to set the sizes and thresholds of the various control structures of the *operating system.*
tuning	1. modifying the *tunable parameters* so as to improve system performance. 2. the reconfiguration of the *operating system* to incorporate the modifications into *executable* version of the system.
userid	an integer value, usually associated with a *login name;* the userid of a *process* becomes the *owner* of files *created* by the process and descendent (*forked*) processes.
utility, utility program	a standard, generally useful, permanently available *program.*
version	a separate *program* product, based on an existing one, but containing significant new code or new functions.

virtual memory	see *memory*.
VTOC	Volume Table Of Contents is the section of a disk which shows how the *partitions* on the *disk* are allocated.

Index

307-303

UNIX
PRESS

A Prentice Hall Title

ISBN 0-13-947086-7